sRIO SERIAL BUFFER FLOW-CONTROL DEVICE

Advanced Datasheet
80KSBR200

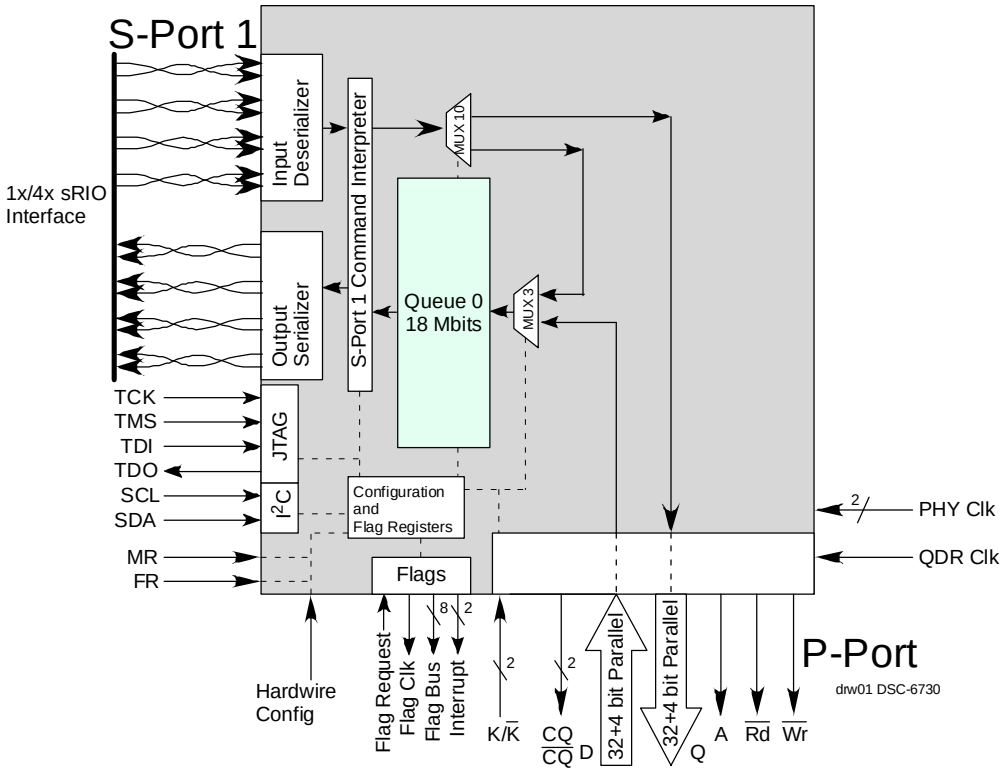
Device Overview

The IDT80KSBR200 is a high speed Serial Buffer (SerB) that can connect to any Serial RapidIO compliant interface. This device is built to work with any sRIO device and especially with the IDT Pre-Processing Switch (PPS), IDT70K200. The SerB performs buffering and off-loading of data as well as buffer-delay of data samples in various environments. This device primarily acts as an master in which the SerB bursts data to a programmed memory location once some criteria have been meet. This combination of storage and flexibility make it the perfect buffering solution for sRIO systems.

Features

- **Serial RapidIO Port**
- **Interface - sRIO**
  - *One four-lane (4x) link, configurable to one-lane (1x) link*
  - *Port Speeds selectable: 3.125 Gbps, 2.5 Gbps, or 1.25 Gbps*
  - *Short haul or long haul reach for each PHY speed*
  - *Support 8-bit and 16-bit deviceID*
  - *Error management supports standard*
  - *sRIO version 1.3*
  - *Class 1+ End Point Device*
- **10 Gbps Throughput**
- **18Mbit Internal Density**
- **Programmable Target Address**
- **Packet Tally Indicator**
- **Packet Interval Timer**
- **Replace Missing Packet**
- **Optional External QDR SRAM Available**
  - *Up to 72Mbit external QDR SRAM*
  - *QDR SRAM, 200 MHz; (18M, 36M, 72M)*
- **Seamless Integration of Internal and External Memory**
  - *Internal and external memory functions as a single buffer*
- **Single Port Buffering**
- **Status Flags for Combined Internal/External Memories**
  - *Full, Empty, Partially Empty, Partially Full*
- **Direct or polled operation of flag status bus**
- **Optional Watermark**
  - *Serial Buffer can Either Send a Flag or Transmit Data at a Specific Packet Count*
- **Interface - I^2C Interface Port**
  - *One I^2C port for maintenance and error reporting*
- **Interface - JTAG Interface**
  - *JTAG Functionality for boundary scan and programming*
- **High-Speed CMOS Technology**
  - *1.2V Core operation with 3.3/2.5V JTAG interface*
- **Package: 484-pin Plastic Ball Grid Array**
  - *23mm x 23mm, 1.0mm ball pitch*

Block Diagram



Notes

Notes

Table of Contents

Notes

Notes

List of Tables

Notes

Notes

List of Figures

Notes

Notes

1.0 Functional Description

The IDT80KSBR200 is a Serial RapidIO™ sequential buffer (SerB) flow-control device consisting of up to 18Mbits of on-chip memory with expansion of one QDR SRAM externally bringing the total buffering capacity to 90Mbits of storage. This device is built to work with any sRIO device and especially with the IDT Pre-Processing Switch (PPS) number IDT70K200.

In this configuration, the main application is working in conjunction with the PPS. In applications were multiple DPSs are used with the PPS, the SerB can function as an over-flow port to handle traffic that is block on any given port or, as a delay buffer to store data and present it at a later time. This is important in DPS applications were time samples are compared with the previous sample such as Cellular Base Stations. Please refer to the application note “Serial Buffer and Pre-Processing Switch”.

The SerB fully complies to the sRIO specification version 1.3 and is implemented to a class 1+ end-point device.

This device operates as a master. In the sRIO environment, a master is defined as a device that originates data transfers, either to or from that device. A slave is one that responds to commands from other devices to move data. As a master, the SerB can receive data and at a pre-programmed water level (number of packets), the device will form and transmit either packets or status (e.g., doorbells) to a programmed location.

The SerB performs buffering and off-loading of data as well as buffer-delay of data samples in various environments. This device can act as a master in which the SerB writes data to a programmed location once the criteria have been meet. This combination of storage and flexibility make it the perfect buffering solution for sRIO systems.

For applications requiring larger buffers, an additional 72Mbits of QDR SRAM can be attached via the Parallel Port. The two memories are seamlessly connected by the Serial Buffer to form a large, 90 Mbit buffer memory. The QDR SRAM interface runs at speeds of only 156.25MHz allowing lower cost memories to be used as well as easier board layout. Data rates still support up to 10Gbits/s (OC-192) thoughput in the device to maintain full sRIO four-lane compliance.

The device provides Full flag and Empty flag status for the queue for write and read operations respectively. Also a Programmable Almost Full and Almost Empty flag for the queue is provided.

A JTAG test port is provided running at 3.3V, device has a fully functional Boundary Scan feature, compliant with IEEE 1149.1 Standard Test Access Port and Boundary Scan Architecture. The SerB can also be programmed via the JTAG port. There is also an I^2C processor port for programming and retrieving information from the configuration registers.

The device is configured into a single queue comprising the full internal memory and potentially the external memory if attached. The device treats the full amount of memory, wether internal or a combination of internal and external, as a single memory block. Status flags from that queue, either referring to the writes (full flags) or the reads (empty flags) to or from that queue represent the total amount of memory. Flags can be read from the serial port or from the I^2C or JTAG port. Proactive flags can be configured to send a doorbell and/or change the interrupt pin once a flag is set. Partial full and empty flags can be programmed to provide reaction time for writes and reads respectively. Flags associated with reaching water marks are available in addition to the full and empty flags.

Further information regarding this device and follow-on devices with added functionality are available from IDT.

Notes

1.1 Interface Overview

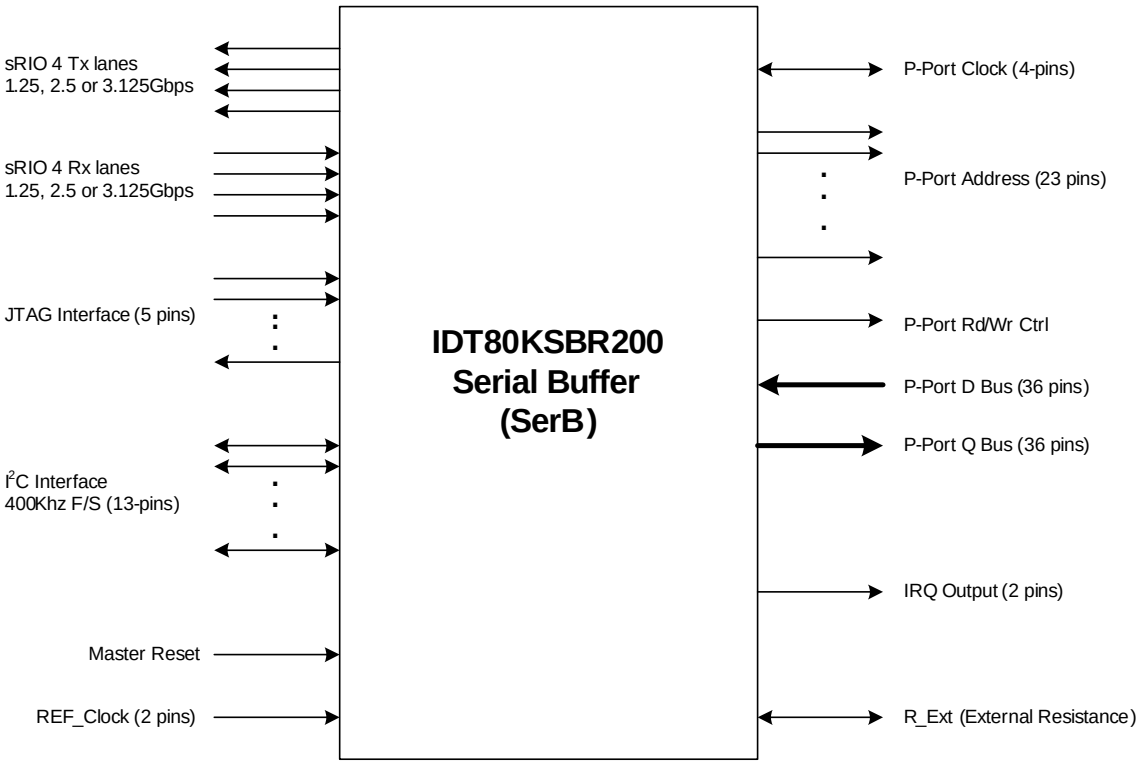


Figure 1 Diagram of SerB Interfaces

1.1.1 sRIO Port

The sRIO interface is the main communication port on the chip. This port is compliant with the serial RapidIO™ v. 1.3 specifications. Please refer to the serial RapidIO™ specifications for full detail.

There are 4 uni-directional differential links for a total of 8 pins. Each can run at 1.25, 2.5, or 3.125Gbps programmable. Both sRIO data (sample) and maintenance packets are transmitted and received on these links.

1.1.2 Parallel Port

P-Port interface is used as a memory expansion port. As a memory expansion port, one of the designated QDR SRAM devices can be connected. If P-Port is connected to one of the designated SRAM devices, it will maintain the clocking and full interconnection to drive the SRAM device.

1.1.3 I2C Bus

This interface may be used as an alternative to the standard sRIO or JTAG ports to program the chip and to check the status of registers - including the error reporting registers. It is fully compliant with the I2C specification, It has 13 pins and supports both Fast- and Slow-mode buses [1]. Refer to the "I2C" chapter for full detail.

1.1.4 JTAG TAP Port

This TAP interface is IEEE1149.1 (JTAG) and 1149.6 (AC Extest) compliant [10, 11]. It may also be used as an alternative to the standard sRIO or I2C ports to program the chip and to check the status of registers - including the error reporting registers. It has 5 pins. Refer to the JTAG chapter for full detail.

Notes

1.1.5 Interrupt ($\overline{IRQ}$)

An interrupt output is provided in support of Error Management functionality. This output may be used to flag a host processor in the event of error conditions within the device. Refer to the Error Management section for full detail.

1.1.6 Reset

A single Reset pin is used for full reset of the SerB, including setting all registers to power-up defaults. Refer to the Reset & Initialization section for full detail.

1.1.7 Clock

The single system clock (REF_CLK+ / -) is a 156.25 MHz differential clock input. Refer to the Clock section for full detail.

1.1.8 R-Ext (Rextn & Rextp)

These pins are used to establish the drive bias on the SERDES output. An external bias resistor is required. The two pins must be connected to one another with a 12k Ohm resistor. This provides CML driver stability across process and temperature.

1.1.9 SPD[1:0]

Speed Select Pins. These pins define the sRIO port speed at RESET. The RESET setting may be overridden by subsequent programming of the Serial Port Configuration Register. SPD[1:0] = {00 = 1.25G, 01 = 2.5G, 10 = 3.125G, 11 = RESERVED}. These pins must remain STATICALLY BIASED after power-up.

Notes

Notes

2.0 Application

2.1 PPS Data Storage

The SerB's primary application is for a Basestations using the IDT's Pre-Processing Switch (PPS). The SerB will be a storage device, holding large amounts of data passed to it by the PPS and with all of its internal memory allocated to queue 0. In this application, the S-Port on the SerB will connect to one of the 4x ports of the PPS. The PPS will pass approximately 10ms of data to the SerB at which time the SerB will start to pass it back to the PPS as a multicast. It is expected that the data flow will remain constant with 10ms (or other designated quantity) worth of data always in storage. The Basestation uses the data for decryption purposes.

The following are items of note concerning the PPS application:

- The SerB has the ability to act as a simple master.
  - The SerB's application with the PPS will be to broadcast data. It must be a master to perform a broadcast, even if the data is requested.
  - The SerB has the ability to initiate writes. Mainly to prevent overflow and to perform broadcasts when waterlevel is reached (timed event). This avoids requiring the DSP to increase congestion by requesting data and controlling the SerB.
- The SerB will typically perform SWRITEs.
  - The target address(s) generated by the SerB is programmable.
  - The packets are stored in the format they come in and are rebroadcast with simple changes to the headers
- The DSPs have the ability to read the SerB registers through the PPS.
  - The DSP may send a maintenance read/write packets to the SerB requesting register information.

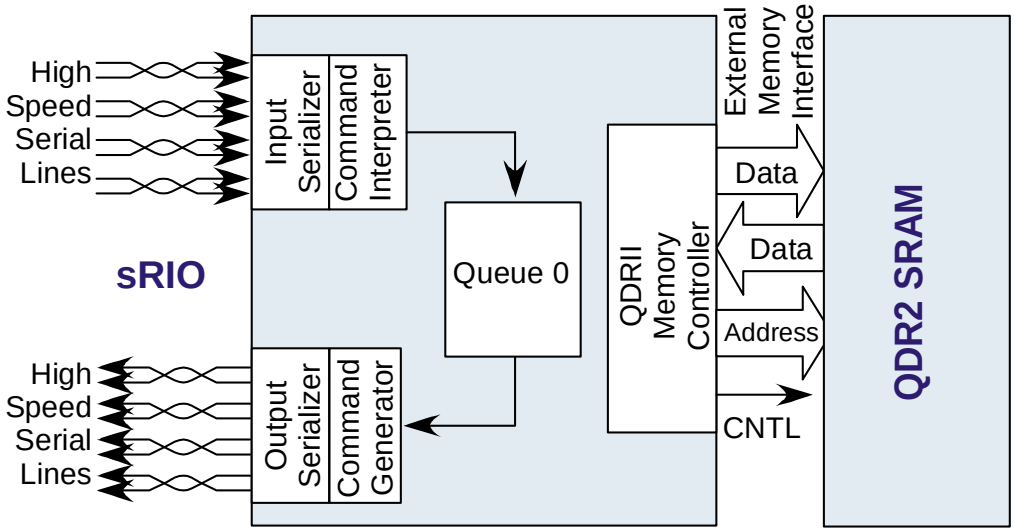


Figure 2 PPS Data Storage

2.2 Compatible External Memories

The P-Port, as a FIFO controller shall connect to an external memory device. There are two designated memory devices, which may be connected to the SerB. These are:

- QDRII-B4 SRAM with 36-bit bus in 36M size
- QDRII-B4 SRAM with 36-bit bus in 72M size

Only one memory may be connected to the P-Port at a time. **Initial release of the SerB will only support 72M density and support of other devices listed above to follow with subsequent release**. Expansion is available only through increased memory size.

2.2.1 Memory Default Configurations:

The memory default configuration on power up or hard reset is as follows:

Notes

- No external memory is allocated, regardless of whether external memory is present.
- All SRAM is allocated to queue 0
- P-Port outputs default to valid states to prevent possible damage to external devices, unless P-Port is physically disabled by the external pin.

Notes

3.0 Protocol

The SerB is a packet-handling device. The SerB may be configured to require all packets to be acknowledged, and hold all packets available for retransmission until acknowledgement is received. Incoming corrupted packets are dropped and issues a retransmission request. The negotiation for acknowledgement, retransmission and dropping packets is handled at the local interface level, without intervention of higher-level authorities. The SerB does not forward any packet until it is fully received, verified, and acknowledged (if configured to verify).

3.1 SerB Packet Characteristics

3.1.1 Maximum Packet Size

The sRIO specification requires a maximum packet size of 256 bytes plus overhead. The SerB fully complies to this specification.

3.1.2 Interface Packet Buffer Size

The sRIO specification has defined buffer sizes for the transmit and receive buffers. Included in the buffer specification is the requirement to transmit higher priority packets first. Upon transmission failure, and retransmission, the retransmission may be held up and a higher priority packet interjected if one arrives.

3.1.3 Multicast Packets

The SerB has no special multicast capabilities. To perform a multicast, the case scenario should be set up to perform an SWRITE function. The destination ID for the case scenario should be set to a multicast address elsewhere in the system. The SerB shall perform a multicast by sending the SWRITE to the user designated multicast address, along with the data.

Waterlevel multicast in the PPS application is done the same way. When the waterlevel event is triggered, the SerB issues an SWRITE multicast packet to the PPS multicast address. The SWRITE command is generated by the case scenario.

3.2 sRIO Specification

The SerB serial interface is a standard 1x/4x serial port with sRIO capabilities. In the PPS application, the sRIO port act primarily as an sRIO end-point, but will work as a bus master to perform multicast operations.

All the RIO TWG documents can be found on the RapidIO Members website:

http://www.rapidio.org/apps/org/workgroup/twg/documents.php

The following documents are the final version 1.2 specifications, which can be found under the Members Library section, version 1.3 of the specifications will replace these section files when they are approved by the Steering Committee:

RapidIO_Spec.pdf	Part I through part IV of the spec., version 1.2
gsmlspec.pdf	Part V of the spec., version 1.2
serial.book.pdf	Part VI of the spec., version 1.2
inter-op.pdf	Part VII of the spec., version 1.2
errspec.pdf	Part VIII of the spec., version 1.2
errata1.pdf	Errata 1 to version 1.2 of the spec.
fcspec.pdf	Part IX of the spec., version 1.0
system_bringup_spec.pdf	Annex I of the spec.

The version 1.3 files are currently:

IO_logical.pdf	Part 1: Input/Output Logical Specification
msg_logical.pdf	Part 2: Message Passing Logical Specification
cmn_trnspt.pdf	Part 3: Common Transport Specification
parallel_phy.pdf	Part 4: Physical Layer 8/16 LP-LVDS Specification

Notes

gsmlspec.pdf	Part 5: Globally Shared Memory Specification
serial_book.pdf	Part 6: 1x/4x LP-Serial Physical Layer Specification
inter-op.pdf	Part 7: System and Device Inter-operability Spec
errspec.pdf	Part 8: Error Management Extensions Specification
fcspec.pdf	Part 9: Flow Control Logical layer Specification
encapspec.pdf	Part 10: Data Streaming logical Specification
mcspec.pdf	Part 11: Multicast Extensions Specification
sbtg.pdf	Annex 1: Software/System Bring Up Specification

There is a checklist for compliance to version 1.3 of the RIO specification, which will be used to insure proper RIO operation.

3.2.1 RapidIO Spec. Version 1.3

In compliance with the sRIO specification, the port has the ability to connect directly to a 1x/4x sRIO port on the PPS device, or connect to any other sRIO compliant 1x/4x port. This includes the standard lane fail functions where a failure of any lane on a 4x port will force the device into a 1x operation on lane 0 or lane 2. The SerB has no requirement to perform with more than a single 1x port. Restated, S-Port shall be either a 4x port or a 1x port as designated by the configuration or fail mode, but shall never be four 1x ports operating simultaneously.

The RIO specification is a universal specification and all sections do not fully apply to the SerB. Each of the parts of the specification will be listed individually below along with the compliance level for the SerB. Some of the documents are not complete and published. Some are working group showings.

Each chapter is discussed in a separate section below.

Part 1: Input/Output Logical Specification

The SerB device shall abide by this spec.

Part 2: Message Passing Logical Specification

The SerB device shall abide by this spec.

Part 3: Common Transport Specification

The SerB device shall abide by this spec.

Part 4: Physical Layer 8/16 LP-LVDS Specification

The SerB device does not support this spec.

Part 5: Globally Shared Memory Logical Specification

The SerB device does not support this spec.

Part 6: Physical Layer 1x/4x LP-Serial Specification

The SerB shall abide by this spec.

Part 7: System and Device Inter-operability Specification

The SerB device shall comply with the Generic Class Requirements (class 1+).

Part 8: Error Management Extensions Specification

The SerB device shall comply with this spec.

Part 9: Flow Control Logical Layer Specification

The SerB devices does not support this spec.

Part 10: Data Streaming Logical Specification

The PPS device does not support this spec.

Part 11: Multicast Extensions Specification

SerB device shall abide by this spec (do nothing). A multicast for SerB is a simple write to an address.

Notes

Annex I: Software/System Bring Up Specification

Already comply.

Approved Showings

The following documents are approved showings in the TWG. Each of them will be discussed in detail.

04-11-00031.001

Change to the spec of the Serial RapidIO Receiver Sinusoidal Jitter Tolerance Mask. IDT SerDes is tuned to this spec.

3.2.2 Summation of RIO registers

The SerB shall include all registers required by the RIO spec for configuration.

3.2.3 sRIO Priorities

sRIO has two forms of priority. The first is the Standard sRIO priority. The second is the Virtual Channel form of sRIO. There is a bit set in the data stream where VC = 0 designates standard sRIO priorities, while VC = 1 designates virtual channels. The SerB shall not use virtual channels, but pass any virtual channel data as if it were sent with standard priority.

Standard sRIO has four discrete levels of priority (two bits). Added to the priority is the CRF (Critical Request Flow) bit which is a priority distinguishing bit within a priority (LSB), bringing the total number of priority bits to three. High priority packets are always sent before lower priority packets. Low priority packets do not enter the data stream until the high priority packets are exhausted. The SerB ignores the CRF bit.

In virtual channel prioritization, there are three bits that designate the virtual channel. These replace the sRIO two bit priority plus CRF bit. With virtual channels, each channel is allocated a percentage of the total bandwidth. In this application, all channels are allocated some bandwidth regardless of their priority, preventing high priority packets from stealing the entire bandwidth. The SerB shall not support Virtual Channels, but instead will always transmit higher priority packets first.

The sRIO user may transmit data on any priority with little regard to volume of data. For instance if there operating at close to full bandwidth with critical data, but would like to support additional service on an "as bandwidth available" basis, he may be running with most traffic on the higher priorities and limited capacity on low priorities.

The response packet sent in most applications is intended to be sent at one priority level higher than the received packet, which would limit the usage of the top priority to response packets, but it is not guaranteed that the user would not use the highest priority for other data.

3.3 sRIO Simplified Overview

The operation of the sRIO bus is contained in the sRIO specification. The following comments are provided to provide a superficial understanding of the initialization of the interface, without researching the specifications.

3.3.1 sRIO Sync

The sRIO sync is accomplished by the transmitter sending continuous /K28.5/ codes (commas) on each lane until sync is accomplished. The state machine is shown on page VI-58 of the Physical Layer x1/x4 LP-Serial Specification for RapidIO. The sync is tolerant of occasional /INVALID/ code groups as shown in the state machine and will increase or decrease level of sync, based upon the error level of the interface. Upon completion of sync, each serial lane should be able to successfully transmit and receive 8B/10B codes.

3.3.2 sRIO Alignment

After sync, the lane alignment must be completed. This is accomplished by sending continuous /A/s on all lanes. The /A/s are counted until lane alignment is accomplished. The state machine for the "A" counters is shown on page VI-60 of the LP-Serial Specification for RapidIO. The state machine is tolerant of an occasional /INVALID/ code group, and will increase or decrease the state of alignment (NOT_ALIGNED to ALIGNED_3) based upon the successful transfer of /A/'s on the lanes. A fully successful alignment would enable the 4x mode of sRIO. If links are broken and/or alignment is not possible, the interface will be required to operate with a single link (lane 0 or 2).

Notes

3.3.3 sRIO Mode Initialization

Once sync and alignment is accomplished, the sRIO controlling device will search for the SerB. The steps of the search include SILENT, SEEK, and then DISCOVERY. Once DISCOVERY is complete, the mode will be set to 4X_MODE (optimum performance), 1X_MODE_LANE0, or 1X_MODE_LANE2, depending upon the success of establishing the link. The state machine for the MODE is shown on page VI-64 of the LP-Serial Specification for RapidIO.

3.3.4 sRIO Control Symbols

sRIO requires the transmission of control symbols providing link status every 819.2ns or less whenever the link is otherwise idle. The control symbols are described in section 5.2 of the LP-Serial Specification for RapidIO. These include delimiters /K28.3/ if a packet delimiter is included or /K28.0/ if there is no packet delimiter.

3.3.5 sRIO End-to-End Retransmissions

As an sRIO bus endpoint, the SerB supports end-to-end sRIO retransmissions. This is required for the SerB to meet the sRIO compliance testing as an endpoint. When S-Port is acting as an sRIO slave, the SerB fully acknowledges all link-to-link transactions and end-to-end transactions per the sRIO specification.

As an sRIO bus master, as would be the case with a waterlevel or doorbell master, the SerB has limited capabilities. At the link level, the SerB has the ability to receive acknowledgement of all transactions at the link level and perform retransmissions of any packets for which a retransmission has been requested.

The SerB does not have the ability to support end-to-end retransmissions as a bus master. When a packet is sent out from the SerB as a bus master, an end-to-end response packet should be received in due time. The packet will be handled as follows:

- If the response is an acknowledgement -- the response will be ignored.
- If the response is a retransmission request - a flag will be set and the packet otherwise ignored. No retransmission will be attempted.
- If there is no response - the SerB will not realize there was no response, because it was not looking for one.

3.3.6 The SerB as an sRIO System Host

The SerB has no ability to act as a host in an sRIO system. The SerB does have the ability to act as a bus master on occasion and will take control of the bus to accomplish the transmission of selected data items or perform selected functions. The SerB does not have the ability to control a system or fully interact and interpret the actions of other devices in the system. Bus mastering is limited to the transmission of the designated data.

3.4 The sRIO Packet

sRIO has a defined packet structure for each type of packet. The sRIO specification should be referenced for a complete description of sRIO packets and their architecture. Packet aspects that are significant in the SerB are described here for clarity, but the sRIO specification overrides in the event of a discrepancy.

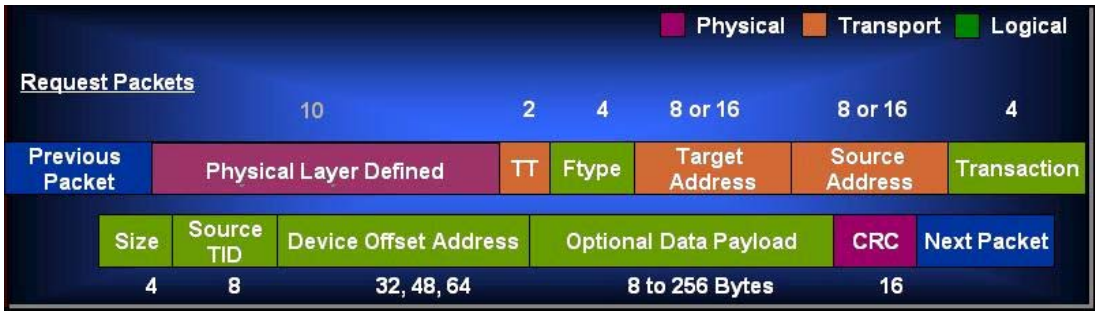


Figure 3 Generic sRIO Request Packet

Looking at Figure 3, the sRIO packet contains the following items:

- Physical Layer Defined header, shown in Figure 4.
- The transaction type, TT, that defines 8 or 16 bit device ID fields, shown in Figure 5.

Notes

- The Ftype, which defines the type of packet being sent. The types are shown in section 3.4.4.
- The Target Address, a.k.a Destination ID. This will be 8 bits or 16 bits, depending upon the state of TT.
- The Source Address, a.k.a Source ID. This will be 8 bits or 16 bits, depending upon the state of TT.
- The Transaction, which is dependent upon the packet Ftype. The supported transactions are described individually.

3.4.1 sRIO Physical Layer Header

The sRIO physical layer header is shown in Figure 14. The various fields are defined in the sRIO Physical Layer 1x/4x specification. The sRIO priority is the priority of the packet during transmission. The contents of the physical layer do not go beyond the interface, except the packet priority (Pri) may be dictated for any transmitted packet. In the SerB, there are two methods for setting the priority.

- If a transmitted packet is a response to a received packet, the sRIO response priority will be one priority level higher than the priority of the request packet, up to the maximum priority.
- If the transmitted packet is being initiated by the SerB, the priority of the packet will be dictated by the SerB. In most cases, the priority will be dictated by the "Case Scenario".

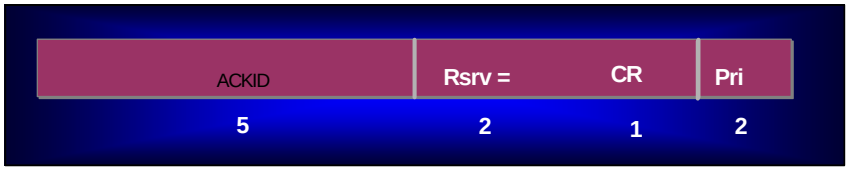


Figure 4 sRIO Physical Layer Header

3.4.2 sRIO Physical Layer CRC

CRC-16 accompanies all sRIO packets and is defined in the sRIO Physical Layer 1x/4x specification. The location of CRC within the packet is shown in Figure 3.

3.4.3 sRIO Transport Layer Header (8/16 bit Device IDs)

During sRIO "bring up", the SerB shall support both 8 and 16 bit device ID fields. Once configured as either 8 or 16 bit, the SerB does not support the other type and will drop packets once configured.

Considering that the only packet type supported is the type configured, the TT bits within the packet are not useful. The SerB insures that the proper TT bits are included in every packet sent. Incoming packet TT bits are a "don't care".

Within the sRIO packet, the TT (transaction type) is used to identify the size of the fields as shown in Figure 5.

TT	Definition
00	8-Bit Device ID Fields
01	16-Bit Device ID Fields
10	Reserved
11	Reserved

Figure 5 Transaction Types (8 or 16)

The source and destination IDs in the sRIO packet will be either 8 or 16 bit as configured. Every sRIO packet that the SerB generates contains a Target ID that has been generated from one of following ways:

- The packet is in response to a request. The Target ID is the source ID of the requestor.
- The packet is generated by the SerB through a "case scenario". The Target ID is included in the case scenario.
- Any packet that is generated by a case scenario will use the Source ID of the queue to send the packet.
- Any flag associated with a queue will use the Source ID of the queue to send the doorbell.

Notes

- Any packet that is generated by the SerB that is unrelated to a particular queue (such as a link error) will use the device ID of the SerB as the source ID.

3.4.4 sRIO Request Packet Types (Ftype 0 - 11)

Within the sRIO specification, 16 packet types may be formed. Packet types "Type 0" through "Type 11" are Request packet types. Packet types, "Type 12" through "Type 15" are Response packet types. Many of the packet types are reserved.

The SerB has limited sRIO functionality, but should be able to imitate any type of command. The SerB initiates commands through the Case Scenario. Case Scenarios have the ability to initiate any type of command by simply entering the correct Ftype and the rest of the sRIO header as desired. The required data may be appended as needed.

The SerB fully supports only selected sRIO commands. The user needs to be aware of the limited SerB functionality, but may be able to pass commands outside the SerB limits if the usage and expectation of the commands fits within the limits of SerB capabilities.

Following are the sRIO commands SerB is capable of supporting:

- SWRITE (type 6)
- CAR/CSR (type 8)
- DOORBELLS (type 10)
- MESSAGES (type 11) (no defined message)

Following are the sRIO commands supported in next phase of SerB:

- NREAD (type 2)
- NWRITE (type 5)
- NWRITE_R (type 5)

The packet types are described in the *RapidIO Interconnect Specification, Part 1: Input/Output Logical Specification* in chapter 4. The following is a list of the packet types and the level of support the lite protocols shall offer.

Type 0 Packet Format (Implementation Defined)

Type 0 packets shall not be used on the SerB.

Type 1 Packet Format (Reserved)

Type 1 packets are not defined in the sRIO spec and shall not be used in the SerB. If received, they are simply passed unaltered at the logical level.

Type 2 Packet Format (Request Class)

Type 2 packets are described in section 4.1.5 of the sRIO spec. Type 2 is used for NREAD and ATOMIC in standard sRIO. The SerB does not support neither NREAD nor ATOMIC packet format.

Type 3-4 Packet Format (Reserved)

Type 3 and Type 4 packets are not defined in the sRIO spec and shall not be used in the SerB. If received, on the sRIO port with an SerB destination ID, an error message shall be sent. When a case scenario is loaded with type 3 or 4, the type shall be passed along with any data. No further interpretation should be needed.

Type 5 Packet Format (Write Class)

Type 5 packets are described in section 4.1.7 of the sRIO spec. Type 5 is used for NWRITE, NWRITE_R, and ATOMIC in standard sRIO. As with Type 2 packets, the priority must be identified so it can be passed.

Type 6 Packet Format (Streaming-Write Class)

Type 6 packets are described in section 4.1.8 of the sRIO spec. Type 6 has only one function (SWRITE), which is limited in scope with no response needed. Therefore, the entire SWRITE packet must be passed unaltered, except for the addition of a priority designation.

The PPSc generates SWRITE packets, so the primary packet the SerB will see in PPS applications is SWRITE. The SerB must accept SWRITE packets as they are received, because the PPS has no backpressure mechanism and a delay in packet acceptance will mean packet loss.

Notes

Type 7 Packet Format (Reserved)

Type 7 packets are not defined in the sRIO spec and shall not be used in the SerB. If received, on the sRIO port with an SerB destination ID, an error message shall be sent. When a case scenario is loaded with type 7, the type shall be passed along with any data. No further interpretation should be needed.

Type 8 Packet Format (Maintenance Class)

Type 8 packets are described in section 4.1.10 of the sRIO Input/Output Logical Specification. These packets are the CARs and CSRs necessary for programming and reading the status/capability of the SerB. The SerB must fully support type 8 packets.

Type 9 Packet Format (Reserved)

Type 9 packets are not defined in the sRIO spec and shall not be used in the SerB. If received, on the sRIO port with an SerB destination ID, an error message shall be sent. When a case scenario is loaded with type 9, the type shall be passed along with any data. No further interpretation should be needed.

Type 10 Packet Format (Doorbells)

Doorbells are not defined in the sRIO, Part 1, "Input/Output Logical Specification", but are listed as "reserved" in section 4.1.11 of that spec. The Type 10 packets are defined in the Part 2, "Message Passing Logical Specification". The SerB shall issue doorbells as defined in section below.

Type 11 Packet Format (Messages)

There is no use identified for type 11 packets. These packets normally carry non-doorbell messages. These packets are also considered "reserved" in the "Input/Output Logical Specification", but are defined in the "Message Passing Logical Specification".

3.4.5 sRIO Response Packet Types (Ftype 12 - 15)

Within the sRIO specification, packet types, "Type 12" through "Type 15" are Response packet types. Of the response type packets, all are reserved except packet type 13, which will be used for all response packets. If a response packet is received with a type other than Ftype 13, the packet shall be ignored and an error flagged.

Type 13 Packet Format (Response)

Ftype 13 packets are defined in the sRIO Part 1, "Input/Output Logical Specification" in section 4.2.3. The SerB fully supports Ftype 13 packets.

3.4.6 sRIO Transaction IDs

Every sRIO transaction must have an ID that cannot repeat itself within a designated time. That designated time is the time that a packet may remain alive, including all blockages, retransmissions and acknowledgements. In the case of the SerB, retransmission capabilities beyond the link level are not supported, and therefore the transaction ID is not used.

Regardless, the SerB must handle incoming transaction IDs and generate outgoing transaction IDs. The SerB will attempt to categorize outgoing transaction IDs. Within the SerB there are multiple sources of packets, where a queue may be generating packets, plus the device itself may generate packets. In some cases, one part of the device may not know what other parts are doing.

Source ID, Destination ID, and Transaction ID all are used to identify a unique packet. In addition, response packets are identified as a "response". Using all of these identifying markers guarantees that the SerB is not capable of generating a transaction ID that would interfere with those generated by another entity.

The following items describe the use of transaction IDs within the SerB.

- Incoming transaction IDs will be returned with any response packets. This includes any response messages, responses to NWRITE_R and other packets that require responses.
- Every transaction generating portion of the SerB will have it's own unique block of transaction IDs to loop on.
- The transaction ID includes the source ID of the transaction, so we will not be interfering with other devices in the system generating transaction IDs.
- There will be 32 transaction IDs allocated to every sRIO packet generating entity within the SerB. The doorbells will be allowed more, since there may be more active at a time.

Notes

The transaction IDs are allocated per Figure 6.

ID Range	sRIO Packet Generating Entity
31-0	Queue 0 Output
127-32	Reserved for future use
159-128	Device ID of the SerB
255-160	Doorbells and reserved

Figure 6 Transaction ID Range for sRIO Packet Generating Entities

Source ID, Destination ID, and Transaction ID all are used to identify a unique packet. If this includes the transaction type or some additional ID, the problems of multiple identical transaction types would be solved. Response packets are identified as response packets, which identify the originator of the request, the source ID of the responder and the transaction ID. Combining these identifies a unique packet despite the possibility of otherwise conflicting transaction IDs.

3.4.7 sRIO Packet Blockage and Priorities

The SerB is not a switch and should not be involved in blocking packets. Despite this, the SerB may be unable to transmit packets or packets may be blocked by downstream devices, requiring the SerB to hold and retransmit packets. When connected to the PPS, the SerB should not be reordering packets based upon priority, because packets are issued based upon time in the buffer and not priority.

When the SerB is used in non-PPS applications, it may be necessary to transmit packets based upon priority. Blocked packets would be held and transmitted after subsequently received higher priority packets have been transmitted. In this situation, blockage may develop if the inflow to the SerB exceeds or equals the outflow. Typically higher priority packets would be initiated for command and symbol passing.

3.4.8 The sRIO Write Packet, Type 5, Special Considerations

The SerB may receive and issue both type 5, NWRITE and NWRITE_R packets. The SerB has no ability to issue or receive any of the three ATOMIC packets. The wrsize accompanying the data will be stored as part of the packet header in the SerB to allow correct identification of the packet length for subsequent transmission of the packet as the packet leaves the SerB.

sRIO Type 5 packets assume the recipient device is addressable as a side address memory. The SerB is a FIFO and will store the data seqentially and transmit data sequentially, regardless of the address accompanying the data. The address will be stored as part of the packet header in the SerB, and may be used when the packet is again transmitted.

Despite not using the addresses for data storage, the addresses are used in some applications to detect missing packets.

3.4.9 The sRIO Maintenance Packet, Type 8, Special Considerations

The sRIO Maintenance Packet is a Type 8 packet and is used for programming and/or reading the CARs and CSRs. In addition, the Port-write maintenance packet may be generated as an error response as defined by the sRIO Error Management Specification.

The sRIO Maintenance Packet allows in-band control of the SerB configuration. The RIO specifications define a number of registers for end-point devices, which is described in the Register section.

sRIO maintenance packets are Type 8 packets and have the ftype field set to 1000b. These packets are described in section 4.1.10 of the sRIO input/Output Logical spec. In addition, information on the tt and hop count can be found in section 1.3 of the Common Transport Specification. An example of the structure of a type 8 packet is shown in Figure 7. The configuration registers are all 32 bits or less, and all packets will carry 32 bits regardless of whether all 32 bits are needed.

Notes

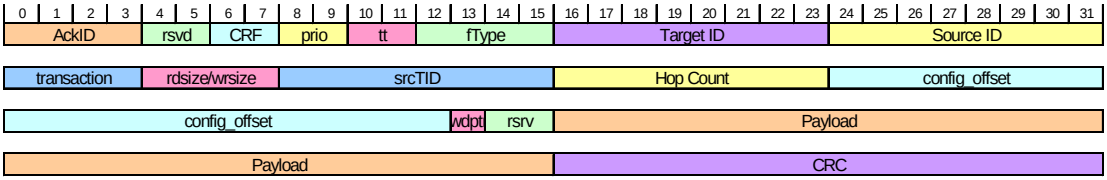


Figure 7 sRIO Maintenance Request Packet (Type 8)

- AckID = Transaction ID for link acknowledgements
- CRF = Critical Request Flow, not used in SerB
- prio = Packet priority
- tt = Transaction type, 00 = 8 bits, 01 = 16 bits
- fType = 1000, for a type 8 maintenance packet
- Target ID = the destination ID of the SerB, is 16 bits if tt = 01
- Source ID = the ID of the sending device, is 16 bits if tt = 01
- transaction = specifies whether request is read, write and/or response, see sect 4.1.10 of sRIO Input/Output Logical spec
- rdsize/wrsize = see sect 4.1.2 of sRIO Input/Output Logical spec
- srcTID = the Transaction ID for sRIO end to end retransmissions
- Hop Count = Not important to an end point.
- config_offset = the configuration register address
- wdptr = part of rdsize/wrsize
- Payload = 32 bits of data destined to be written to the designated register
- CRC = 16 bits of CRC

The sRIO maintenance request packet will receive a response packet as shown in Figure 8. The response will be returned to the sender of the request and include a "status" of the request. The status is identified in section 4.1.10 of the sRIO input/Output Logical spec. The SerB shall observe that 0000b indicates "done" and 0111b indicates "error".

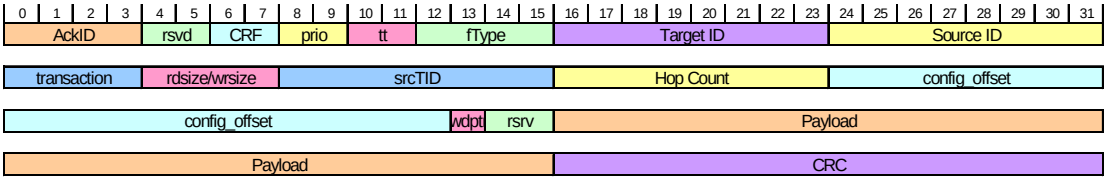


Figure 8 sRIO Maintenance Response Packet (Type 8)

Other than the status field of the packet, the fields serve the same function as the request packet or are unused. Upon a read request, the Payload is the data content of the selected configuration register. When initiating a Maintenance Response Packet, the hop count will be set to 0xFF.

The fields of the response packet are as follows:

- AckID = Transaction ID for link acknowledgements
- CRF = The incoming CRF is returned in the response
- prio = Increased to one higher than the request
- tt = Same as the request
- fType = Same as the request (8)
- Target ID = The Source ID of the request (a simple swap)
- Source ID = The Target ID of the request (a simple swap)
- transaction = specifies whether request is read, write and/or response, see sect 4.1.10 of sRIO Input/Output Logical spec
- status = 0000b means done, 0111b means error

Notes

- srcTID = the Transaction ID for sRIO end to end retransmissions (generated at the interface)
- Hop Count = Set to 0xFF to initiate the hop count
- Payload = 32 bits of data read from the designated register
- CRC = 16 bits of CRC

3.4.10 Virtual Channel Handler

There is no virtual channel handler in the SerB. Virtual channels do not appear beyond the sRIO interface and have no affect on SerB operation.

3.5 sRIO Control Symbols

The sRIO control symbols are described in the sRIO Part 6: 1x/4x LP-Serial Physical Layer Specification in Chapter 3. Of particular note, these symbols are used to acknowledge all sRIO packets. The SerB shall support the following Stype 0 control symbols.

- Packet Accepted
- Packet Retry
- Packet Not Accepted
- Status
- Link Response

These control symbols shall be used to acknowledge all incoming sRIO packets and doorbells. Outgoing packets and doorbells shall expect a response and report errors when they occur.

3.6 Use of CRC and CRC Errors

The SerB shall have the capability of using CRC-16 and is defined in the sRIO "1x/4x LP-Serial Physical Layer Specification" in section 2.4.2.

The following rules dictate uses of CRC within the SerB:

- CRC will be CRC-16 with two bytes in size.
- CRC errors shall be counted. The counts shall be stored and readable through the configuration registers.
- If retransmission is turned off, a packet with CRC errors shall be dropped. There is no indication a bad packet was received. The CRC error will be logged. The user may use higher level detection to retransmit a section of data.
- All CRC errors will set the error flag and may cause interrupts or doorbells per the flag configuration.
- sRIO contains CRC in all packets. CRC suppression is used with the PPS.
- The minimum packet size when retransmit is turned on is 8 bytes payload.

3.7 Parallel Port Interface

The P-Port is a standard parallel interface that is used to drive QDRII SRAM devices. It has a 36-bit data bus, and other control signals that may be connected to a standard QDRII memory interface.

The SerB parallel port options:

- The SerB may act as a FIFO controller, using an external QDRII-B4 x36 memory as extra storage space that may be allocated to the internal FIFO queue as desired.
- P-Port may be disabled, either by a pin, or by programming an internal register.

The definition of the P-Port interface in this specification is guidance only. The overriding requirement is that the SerB must connect to a QDRII-B4, x36 device.

Notes

4.0 Data Handling Within the SerB

The S-Port on SerB has the ability to act as an sRIO Endpoint or as an sRIO Bus Master. When the SerB is outputting to an sRIO port, the queue holds the output packet routing information designating the final destination for the data.

In the PPS application, the SerB will typically act as an sRIO endpoint (slave), and will respond to commands received through the PPS. In the event that there is an active waterlevel = watermark, the SerB shall become the sRIO bus master to send the multicast packet to the PPS.

4.1 Inputting Data to the Queue

Incoming serial data must be directed to a queue upon entry into the SerB. The incoming packet data carries an identifier that selects a "case scenario" for the data that includes the routing information. In sRIO, the source ID of the data selects the "case scenario" for the data. This is designated at "**Case Scenario Mode**".

4.1.1 Command Input Buffer

There is an input buffer on the SerB that is capable of stacking a small number of commands. There is a separate buffer for read and write commands. It should be noted that commands may become blocked by activity within a queue in the SerB.

4.1.2 Output Buffer

There is an output buffer that is capable of stacking output packets on the SerB. In the event that multiple output packets become stacked within the buffer, the higher priority packets will be sent first. A packet that is blocked on the bus for any reason, will prevent the transmission of subsequent same priority packets and lower priority packets until the blocked packet successfully sends, or is discarded.

4.1.3 Writing More Data than can Accept

Whenever an attempt is made to write more data to queue than there is space available to accept, the SerB will go through the following stages:

- When the queue is full, a Full Flag will be set. The flag may then send any interrupts or doorbells to unmasked locations.
- The incoming data will be accepted in full packets and fill the input buffer on the FIFO port.
- If the input buffer contains data that it cannot flush into the queue, the data will sit there, preventing the port from writing to the queue. Priority and maintenance packets will not be blocked, but data packets will be blocked
- If the input buffer also overflows, the incoming packets will be rejected. Only full packets will be accepted. If there is not room to store a complete packet, the entire packet will be rejected. The sender will be notified of the packet rejection.
- Once the full queue empties enough to allow the data in the input buffer to flow into the queue, the input buffer will again be free to accept more data.

Space Available

The Space Available flag is located in the Full Flag register. It is assumed that if multiple sources are writing to the SerB, they will poll the space available register to see how much room is available for writing. When the space available flag toggles, the flag will be sent to the destination ID within the register and to the port designated by the mask registers. Any multicast will be the responsibility of the user.

4.2 Outputting Data from the Queues

The queue output is dedicated to a port and cannot be reconfigured. The queue is configured with a "Case Scenario" that dictates a destination to which the data is sent. The sending of data is triggered by a waterlevel (event). The configuration registers are used to set up the output mode.

Notes

4.2.1 Burst Write Start/Stop Address

The SerB has the ability to pass large quantities of data with minimal overhead. Data can be passed from sRIO to down stream RIO system memory address as either an SWRITE or NWRITE type packets. To start the data burst, the starting sRIO memory address should be loaded into the ***Case Scenario Start Address Register***, along with the ***Case Scenario Stop Address Register*** and an indication of whether to wrap or stop when hitting the maximum address. ***Case Scenario Next Address Register*** initially starts off with same value as the Start Address and increments by the quantity of data transmitted with every packet until reaching Stop Address. If a doorbell or interrupt is desired, that may also be programmed.

The configuration is "case scenario" based. The start, stop counter, and wrap/stop bits are all configured with the “case” in the configuration register. Therefore any data sent to this case, will increment the counters and addresses checked. It is assumed that the user will be responsible for maintaining data integrity, and will probably use the case for one source of data only.

The SerB will form sRIO packets, append the incrementing memory address and send the data out as an sRIO memory data. The memory addresses will continue to increment with subsequent data until all data is transmitted and the port is reconfigured or the address is reset to a new location.

Stop/Wrap on Memory Write

Once sufficient data has entered the SerB to cause the sRIO memory address to reach the stop address programmed into the configuration register, the SerB will do the following:

- The SerB set the "Mem Stop" bit in the flag register. Unmasked doorbells and interrupts will be sent.
- The case scenario will be checked for the WRAP/STOP bit setting.
  - If stop, the remainder of the packet will be transmitted. Stop condition must be cleared before any more data can be transmitted.
  - If wrap, the address will reset to the start address after the overflow packet is fully transmitted. There will be no attempt to perform the wrap in the middle of a packet. It is the user's responsibility to insure that wrap boundaries concur with packet boundaries.

4.3 Use of Acknowledgements

sRIO has requirements for acknowledgements that must be observed by the SerB and are described in the sRIO specification. Both the ability to enable ACK/NACK and the time-out associated with packet failure may be set by programming the device configuration registers. The receipt of a NACK or the failure to receive an ACK within the allocated time will trigger the retransmission of all packets sent after receipt of the last ACK.

When configured to require packet acknowledgements, the following rules apply:

- Packet is sent with an identifier in the header
- Additional packets may be sent before acknowledgement is received
- Packet identifier is incremented for each packet (and wraps)
- Good packets must be concluded with the End of Good Packet (EGP) marker
- If a known bad packet is sent, it should be marked End of Bad Packet (EBP) marker.
- Once a full packet is received, the receiving device must send an acknowledgement or a rejection notice.
- If sender times out without an acknowledgement, the packet and all subsequent packets are sent again.
- If rejection notice is received, packet must be retransmitted and all subsequent packets are retransmitted.
- Packet is rejected if link errors, CRC errors, or EBP code is received
- If the FIFO fills due to the inability to successfully transmit, it indicates a link down and appropriate flags and priority packets sent (if possible).

Note that link level transmissions require that packet acknowledgements be received in the order sent. If a packet is not acknowledged, or acknowledgements are received out of order, it is necessary to retransmit all packets starting after the last packet for which a valid ACK was received.

sRIO link acknowledgements require acknowledgments in the order packets were transmitted, but end-to-end acknowledgments may be received in any order.

ACK and NACK are performed through link management packets and are not priority packets. ACK and NACK may only be used when "retry-on-error" is enabled.

Notes

4.4 Idles

When the S-Port is not sending packet data, 8B/10B Idles shall be transmitted, along with any link maintenance packets needed per the protocol spec. Idles received, will be ignored and not result in data being stored within the SerB.

4.5 Case Scenarios

The "Case Scenario" is method used to generate the sRIO packet headers when data is transmitted out of the SerB. The case scenario is established to route every sRIO data packet that is originated by the SerB. A single queue may have data intended for several different destinations as defined by the case scenario.

The case scenario may be programmed to be any sRIO command type followed by data, allowing fairly sophisticated command generation with little overhead. While the SerB may program any command into the case scenario, it is not guaranteed that the SerB is capable of fully executing more than the designated command types. The user may be able to use this feature to extend the SerB capabilities.

The "Case Scenario Register" must be programmed before use. Every data packet that leaves the SerB must contain an sRIO packet header. The following are the rules describing "Case Scenario".

- Case Scenario is programmed into the configuration registers.
- Every data packet originated by the SerB must use a case scenario
  - sRIO Response packets do not use case scenarios
  - sRIO Doorbells, messages, and other packets do not use case scenarios
- The queue is programmed to always select a case scenario for all data that leaves that queue.
- The destination ID is used to route the packet to the queue.

4.5.1 sRIO Destination IDs for queuing incoming data

The SerB itself has a device destination ID, and any incoming sRIO packets that do not contain data (e.g. configuration register updates), should use this device destination ID. The device destination ID is further described in the configuration registers section. It is searchable on the sRIO bus and is programmed during the sRIO "bring up".

The queue may be programmed with a destination ID in the configuration register (separate and distinct from the device destination ID). This destination ID is not searchable and not programmed in accordance with the sRIO "bring up" specification. Instead, the register must be programmed using the same methods as most of the other configuration registers. Any data coming over the sRIO port, carrying a destination ID that matches the destination ID for the queue will be loaded into that queue.

The destination ID is an eight bit designation within the sRIO packet header. The destination IDs programmed in the configuration registers are also eight bits. The programmed destination ID will be used as the source ID during sRIO transmission. Figure 9 below shows the location of the destination/target ID and the source ID in a typical sRIO packet.

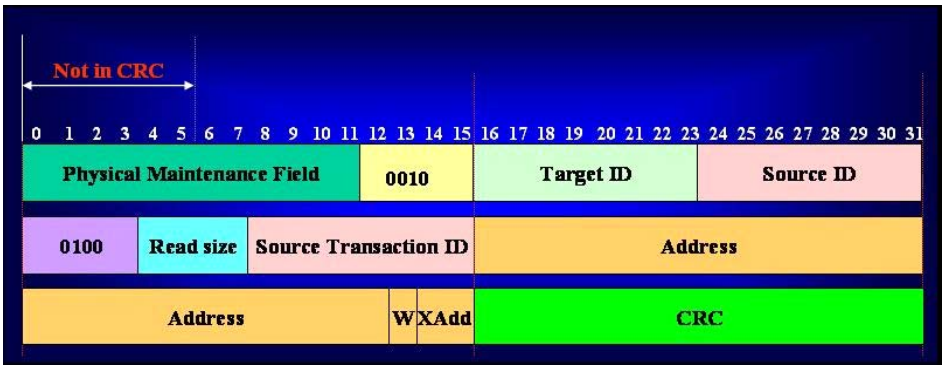


Figure 9 Typical sRIO Packet showing location of Source and Destination IDs

Destination IDs are the means of communication within an sRIO environment. It is required that every sRIO packet have a destination ID and a source ID.

Notes

4.6 Waterlevel and Watermarks

In the SerB, the "watermark" is a programmable event trigger threshold for the data level associated with a queue. The term "waterlevel" refers to the actual data level within a queue, regardless of whether a "watermark" is used. When the waterlevel in the queue reaches the watermark, an event will take place, depending upon the configuration. The waterlevel and watermarks are used primarily as an indicator to control data flow within a queue.

The waterlevels are different than the PAF and PAE flags, because PAE/PAF flags deal in increments of total memory space (1/256th of the queue total) and do not deal with actual data counts. Waterlevels actually count packets or bytes as designated.

4.6.1 Waterlevel Controls

Waterlevels are primarily controlled in the configuration registers. The following items are available for controlling the waterlevels:

- Data in queue is held in packets.
- Waterlevel - A counter that holds the actual data level in a queue. The count will be in packets.
- Watermark - This word holds the trigger point for the waterlevel. When the waterlevel reaches this point, the flag will be set. The flag may cause other events to occur (doorbells, interrupts, etc.)
- A single packet at a time will be sent in their original sizes. Packets will continue to be sent, until the waterlevel drops below the watermark. The remaining data will be held in the queue.
- Space Available - This is a word in the waterlevel register that indicates how much space is remaining in the queue. The space is in packets. The value of the counter is the total capacity of the queue minus the number of bytes already consumed.

It should be noted that when operating as a waterlevel master, the "master mode" only affects the queue output. It is still possible to receive write commands on the queue input.

4.6.2 Example Uses of Waterlevels

There are several possible uses of waterlevels and watermarks. A few of the suggested applications are as follows.

Stable Data Level in Queue

This application allows the steady maintenance of a data level in a queue. As data is received, it is stored in the queue until the data level reaches the watermark. Upon the waterlevel reaching the watermark, for every packet received, the queue will transmit an equivalent data item. The following items set up this scenario:

- The queue is set up to be a master
- The waterlevel is programmed to count packets
- The watermark is set to the desired number of packets to be held within the queue at all times
- The queue sits idle when the packet within the queue is less than the watermark. Packets are received, but not transmitted
- If the waterlevel reaches or exceeds the watermark, the queue will transmit enough packets to bring the waterlevel back below the watermark.

PPS Specific Use of this Scenario

The basestation application that uses the PPS requires that there be a specific timed delay between the SerB input and output packets. The delay is dependent upon the system requirements, but once the system is configured, it remains fixed. It could be any designated delay, but the maximum in the TI DSP application is 10ms. The quantity of packets that would accumulate within the designated time frame would be dependent upon how many RF cards are used in the basestation.

The PPS issues packets to the SerB on a stable time interval, meaning that by using the watermark to designate a quantity of packets, a time interval can be derived from the total. Using the watermark to trigger packet transmissions, the SerB may be used as a programmed packet delay.

In the typical PPS application, all packets will be identical in length and at equal time intervals. Usually the PPS will reform packets to all be equal in size regardless of the number of antennas, but in some (rare) PPS applications that have multiple antennas, it may be possible for the PPS to send packets of various sizes to the SerB. This should cause no prob-

Notes

lems with the waterlevel timer if packet counts are used. It is expected that all antennas will send data at very stable rates, so the combination of two or more sending packets with different sizes will not interfere with the overall timing if the total quantity is adjusted to accommodate the combined larger number.

Packet Ready

This mode may be used to indicate that at least one packet is available in the queue for reading. The flag will toggle, indicating that one or more packets is in the queue. The following is the setup.

- Queue is programmed to be a slave
- Waterlevel is programmed to use packets
- Watermark is set to desired level.
- The flag masks should be programmed to send the desired interrupt or doorbell to the correct recipients
- The flag will toggle whenever the packet count in the queue goes from zero to one. The flag will remain active as long as the packet count equals or exceeds one.

Space Available at the Inn

This is the reverse of the Packet Ready scenario. When feeding data into any of the ports, it may be necessary to know that there is room to accept the packet or more data. There are a couple of ways to for the user to accomplish this:

- Use the same scenario as "Packet Ready" and but set the waterlevel to one full sized packet below the queue size. An interrupt pin could be used as a "space available" pin.
- There is a flag on the Space Available counter to indicate that there is space for one more full sized packet in the queue. This flag could be used as an interrupt to indicate when the space has fallen below the designated quantity (one packet plus some extra to accommodate latency in shutdown). The space available counter is in the waterlevel register.
- Use the PAF and PAE flags to generate an interrupt or doorbell. This would give a more general indication of the space available, while preserving the watermark for other uses.

The Flag of Impending Doom

In this scenario, the watermark may be programmed more accurately than the partial flags and could be used as an almost full flag or an almost empty flag. The flag could be used to indicate to the user that immediate action must be taken to avoid overflow or underflow.

4.7 Missing Packet Detection and Packet Replacement

In the wireless basestation application that uses the PPS, a missing packet can cause havoc to the overall system. To help overcome occasional missing packets, a missing packet detection and replacement can be performed.

There are four configuration registers that are programmed by the user. The registers contain ***Memory Start Address***, ***Current Memory Address***, ***Memory Address Increment***, and ***Memory Stop Address***. It is expected in the PPS application that all packets bound for a single DSP will be equal sized and have equal address increments, allowing the Memory Address Increment to be used to detect incoming missing packets. When a packet comes into the PPS, the PPS may segment the packet into 8 segments. The SerB will detect missing packets through the use of the address field in the packet header.

In the PPS application, it is expected that the user will be performing memory writes through sequential addresses. Missing packets may be detected by insuring that the first packet starts on the Memory Start Address and the address associated with every subsequent address matches the previous packet address plus the Memory Address Increment. In other words, the Current Memory Address plus the Memory Address Increment should be the new Current Memory Address of the next incoming packet. If a packet is missed, the address should match the Current Memory Address plus the Memory Address Increment added twice. Upon failing that, it is assumed that more than one packet was lost, or some serious failure occurred and the flag is set in the flag register. Upon a serious failure, the Current Memory Address in the incoming packet should be loaded into to the Current Memory Address register, and the SerB will attempt to compare the new Current Memory Address plus Memory Address Increment with the subsequent packet address.

Missing packet detection requires the spacing of the addresses to hold at least two packets. It is not expected that missing packet detection will function properly with only one packet available. If two or more packets are missing, the missing packet detection may require the spacing of the minimum and maximum addresses to allow for storage of at least three packets between the addresses.

Notes

4.7.1 One Missing Packet Detected

If only one packet was lost and the packet that arrived is actually the following packet (detected by the memory address), a marker shall be loaded into the queue to indicate that a missing packet was detected. The incoming packet shall also be loaded into the queue. Two packets shall be transmitted from the queue to attempt to regain the timed data flow (observing transmission rate restrictions). No error is noted and no flag is set.

When the missing packet marker reaches the output of the queue, the SerB shall create a dummy packet and transmit it at the time that is designated for the original packet had it not been missing. The DSP receiving the dummy will realize it is a dummy, and can take appropriate action. It follows the normal waterlevel/watermark scheme for transmitting packets.

If the stop address is reached, a flag event has occurred and the appropriate flags will toggle.

4.7.2 Two or More Missing Packets Detected

In the event that two or more packets are missing, no attempt will be made to reconstruct lost packets. The missing 2 error flag shall be set, which may cause additional doorbells and interrupts. The SerB shall continue to load and transmit data normally, hopefully recovering full operation after the system clears itself of defective data.

4.7.3 Missing Packet Detection Summary

The summary for missing packets is as follows:

- The memory address of each incoming packets shall be checked to insure contiguous addresses.
  - The memory increment added to the former memory address tells you what the new address should be.
  - The memory increment does not change in a system, but will be different between systems. Therefore, packets are known length.
- In the event that an address does not match, it is assumed that there is a missing packet. The memory increment will again be added to the current address and checked with the address of the incoming packet.
  - If the addresses match, only one packet is missing
  - If the addresses do not match, two or more packets are missing, or a serious address misalignment has occurred.
- If one packet is missing
  - A missing packet marker is loaded into the queue
  - The incoming packet shall be loaded into the queue
  - Two packets will automatically be transmitted based on the watermark
  - The packet interval timer will limit the transmission rate to match the PPS acceptance rate.
  - When the missing packet location reaches the queue output, a dummy packet (a packet with all zeros in the payload) shall be transmitted to replace the missing packet
- If more than one packet is missing
  - No changes will be made to waterlevels
  - No lost packet markers shall be loaded.
  - No dummy packets shall be sent
  - The “Missing 2 Packet” flag shall be set in the Missing 2 programmable flag register.
  - If the flag is unmasked, a doorbell shall be sent to the destination ID within the register. The content of the doorbell shall be the content designated in the Missing 2 programmable flag register.
  - The Memory Address of the incoming packet will be loaded into the Current Memory Address register to attempt to realign addresses
  - Processing will continue as normal on subsequent packets, allowing the DSP to decide if action is needed
- At the boundary conditions where the memory address exceeds the stop address
  - On wrap, if the memory increment plus the former memory address exceeds the stop address, the new address will be set to the memory start address. No packet is wrapped in the middle, but the next new packet is set to the memory start address on a wrap.
  - If the next packet address does not start at the memory start address, a packet is considered missing and should be replaced.
  - If a second packet fails to match the designated address (start address + increment), the packet will be handled as described above -- "If more than one packet is missing"
  - The Start and Stop range values must be aligned to the increment boundary (a multiple of the increment).
  - There must be enough space in the queue to hold more than one packet for the “missing packet detection” to function.

Notes

4.8 Packet Tally Indicator

In cases where the SerB bursts data to one of the DSPs, the DSP has no way of knowing that it has received the data. The SerB has the ability to send a doorbell indicating that the designated (programmed) number of packets has been sent, so that DSP may act.

Bursting must be done on frame boundaries within the DSP/PPS application so that the DSP receives the doorbell on the frame boundary. In PPS, messages may be passed more quickly than data packets, so the message may arrive prior to the data. To avoid this, the DSP must decide what the delay is through the system for each (data and message). The DSP may then program an offset into the SerB registers and a frame size.

The TI DSP has no ability to analyze the contents of a doorbell, but instead uses 6 bits of the 16 bit data field as a pointer to an interrupt. The pointer is fixed in the DSP, meaning that the pointer must be programmable in the SerB to match the pointer required by the DSP.

Relating this to the SerB, the "case scenario" is used to identify the DSP as a target. Every time a "case scenario" is accessed, the counter within the case scenario shall increment. When the count reaches the maximum programmed for that case, the SerB shall send a doorbell to the destination ID designated in the case scenario, and the count shall reset. The flag register may be used, except the doorbell must be sent regardless of whether previous flags have cleared.

Regarding the "offset", aside from initial power up/reset, it is uncertain what the trigger event is that would require an offset. Therefore, the SerB shall provide an offset to the first frame count after power up and upon any reset that clears data. Since the offset is contained in the case scenario, it may be accessed at any time by any of the programming sources and can be adjusted as needed.

The "Packet Tally Indicator" Frame Size, Frame Offset, Count, and whether to send a doorbell are contained in each case scenario.

4.9 Packet Interval Timer

The PPS and potentially other devices may not have the ability to accept data at an accelerated rate. The PPS processes incoming data as it arrives, limiting the amount of data that can be accepted in a burst. To solve the problem, a "Packet Interval Timer" has been added to the SerB to regulate the spacing between packets going out the port. There is a separate programmable timer for data packets and priority packets, since they take different routes through the PPS. Every time a packet is sent, the timer is reset and then counts down. Another packet of the same type may not be sent until the timer times out.

4.10 Protocol Translation

Through the sRIO port on the SerB, data may be written to or read from the FIFO. The port also has the capability of initiating data transfers (as a master), and writing data out of the port to another location. In addition, SerB control words may be written into the SerB through the port to configure or to read the status of the device.

When using the SerB in two sRIO domains, translation issues arise. It should be noted that the SerB has limited translation capability. Its primary translation function is receiving data, storing data, and subsequently transmitting the data. The ability to pass commands through the SerB is limited. To insure compatibility, there are constraints upon the data. The SerB will handle all link maintenance functions, required responses, retransmissions, and other negotiations.

In the PPS application, the SerB is essentially an sRIO to sRIO translator. The SerB receives data in packet form, stores it, and then transmits it at the designated time on the same port. The incoming packet must match the outgoing packet in size. PPS uses only a designated (programmed) packet size.

Notes

Notes

5.0 Doorbells and Interrupts

Interrupt pins and packetized Doorbells are used to pass interrupts and messages out of the SerB. Outgoing doorbell packets and interrupts are generated by flags.

A flag is considered any event that results in a bit being stored in the "flag register". The content and masks for the flag registers are detailed in **Flag and Flag Mask Register** section. Events at these locations will cause a flag to be stored at the designated location within the flag register as they occur.

In addition to simply residing within the flag register, any flag may cause an interrupt, notifying external devices that a flag event has occurred. This interrupt is considered a "doorbell" and may be issued in one of the following ways:

- External output pin toggling (two pins, each with a mask).
- sRIO Type 10 packets sent over S-Port.

Each flag register has four mask registers designating which flags should cause the associated "doorbell" or interrupt on the port. A violation of any unmasked flag shall cause the designated interrupt to occur. Of the four mask registers, Mask 1 is associated with S-Port and will cause doorbells to be sent. Mask 3 and Mask 4 are not associated with a port and will cause external interrupt pin 0 and interrupt pin 1o toggle respectively. Mask 2 is reserved for future use.

As a default, the flag register mask will not generate any interrupts (full mask). Interrupt generation must be programmed by the mask registers.

In the event that multiple flags toggle, the interrupts/doorbells will be generated based upon the priority programmed in the flag registers. In the event that flags have the same priority, the flags will be handled in the order they occurred. In the event that multiple flags with the same priority toggle simultaneously, the flag with lowest address will have priority over flags with higher addresses.

5.1 Doorbell Characteristics

When a flag causes a doorbell, the doorbell includes the following:

- The register number containing the toggled flag
- The flag number within the register that toggled
- The entire unmasked content of the flag register (flags only)

sRIO doorbells are limited to a 16-bit payload.

5.1.1 sRIO Flag Doorbell Packet

An 8-bit sRIO doorbell packet is shown in Figure 10. The Target ID of the sRIO doorbell is programmed into the flag register causing the doorbell. The Source ID will be the source ID of the doorbell in the SerB. If the doorbell is generated by a queue, the destination ID associated with the queue will be the source ID for the doorbell. If the doorbell is generated by something other than a queue (e.g. a link error), the sRIO generated destination ID of the SerB shall be used as the source ID of the doorbell.

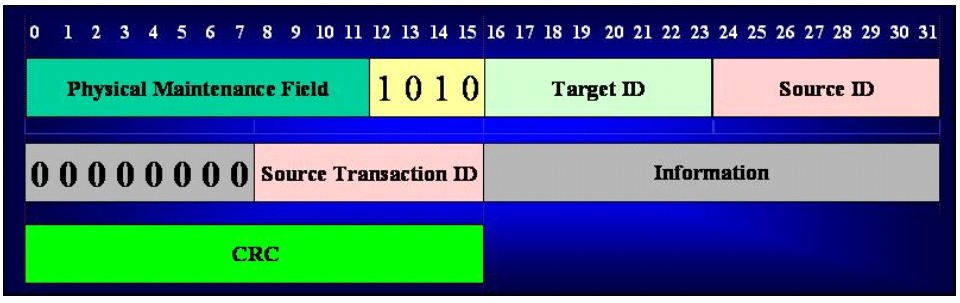


Figure 10 sRIO Doorbell Packet

Notes

sRIO 8/16 Bit Destination IDs for sRIO Doorbells

While typically all transactions with the SerB will be either 8 or 16-bit addresses exclusively, the SerB may be used in systems that mix 8 and 16 bits. Every flag register contains 16 bits that may be used as a destination ID. To define the usage of 8 or 16 bits, every flag register that is capable of sending a doorbell contains a "TT" designation in the flag register that indicates whether the full 16 bits should be used as a destination ID for the doorbell or only the 8 LSBs. The sRIO packet will be formed with the address, based upon the TT bit.

In no case will a transaction contain an 8-bit source ID and a 16-bit destination ID (or the reverse) in the same doorbell. This does not exclude the possibility of a queue using 8 bits as the destination ID for the queue, and then generating a doorbell to a 16-bit destination. It does mean that if a user is trying to mix 8 and 16 bit destination IDs, they will need to correlate the 8 LSBs for both.

sRIO Flag Doorbell Payload

The sRIO Flag Doorbell Payload is a maximum of 16 bits. The payload consists of the following

- 2 bits = Unused
- 6 bits = Register number of the flag that is causing the doorbell
- 8 bits = Register contents showing the status of every flag in the register, regardless of whether the flags are masked.

5.2 External Interrupt Pins Int(0) and Int(1)

Each of the two external interrupt pins may be toggled by any unmasked flag. Each pin has its own flag mask available allowing the user to designate the flag or combination of flags that will cause the interrupt pin to toggle.

There are two types of flags indicated in the flag register, which are RT (Real Time) and CL (Clearable) flags. When an unmasked RT flag toggles an interrupt pin, the pin will remain active as long as the flag is active and cannot be clear, except by reprogramming the mask. When an unmasked CL flag toggles the pin, the user may reset the flag, and the interrupt indication will be removed from the pin until the flag again toggles or the mask is reprogrammed.

It is expected that one of the two pins will be programmed to indicate a generic flag concern, including all flags that may cause concern to the user. The second flag pin would be used to monitor an immediate or frequently used condition, such as "packet ready", meaning that the toggling of the flag generates immediate response without further determination concerning the cause of the interrupt.

There is no ability for sRIO to toggle one of the interrupt pins directly through a command. sRIO may toggle a pin indirectly by creating a condition that causes one of the unmasked flags to toggle, subsequently affecting the designated pin.

5.2.1 Clearing Interrupt

Clearing an interrupt is accomplished by clearing all flags that are causing the interrupt. Since multiple flags are together in a register and additional flags may toggle after a register has been read, completely clearing a register may clear unrecognized flags. The proper usage of flags and how to clear them, is described in section 8.5, Flag and Flag Mask Registers in the notes of this datasheet. As described, writing a "1b" to a flag clears it. Writing a "0b" to a flag does not affect the flag. This way, any flag may be individually cleared or cleared in combination with other flags in the same register.

Notes

6.0 Device Programming

The operational setup of the SerB is accomplished through the programming of the configuration registers. During power up or master reset, the configuration registers default to a known state based upon the configuration established on the hard-wired pins. After power up, the configuration registers may be further altered through programming. It shall be possible to hard-wire the SerB to have full port functionality and be fully programmable through any of the designated programming methods without relying upon a second programming method.

In the priority scheme of configuration, the hard-wired default pin configuration is the dominant configuration during power up or hard reset. The hard-wired inputs will be read on power up or reset, and shall not alter the state of the SerB after completion of power up or reset. The hard-wired configuration may be overwritten through any of the programming schemes, except in a few selected cases (such as designated protocol) where there is no additional programmability.

Once fully powered and hard reset is no longer active, the configuration registers may be reprogrammed or altered by several schemes. The configuration register will retain the last programmed configuration regardless of programming method. One programming method is not dominant over the others, except on Master Reset.

The methods of device programming are as follows:

- Hard wired configuration
- I^2C
- JTAG
- sRIO maintenance packets

The hard-wired configuration will be the initial default setting for the SerB and forced setting after hard reset. The default configurations are shown in the Configuration Register section.

The configuration registers for the SerB are shown in section 8.2 of this datasheet. All configuration registers may be read through I^2C, JTAG, and sRIO protocol priority packets. In addition to the listed configuration registers, there are many registers associated with programming sRIO per the sRIO specification. All bits in the configuration registers are readable by any available method. Bits that have restricted write access may still be read by any method.

6.1 Vendor IDs

For sRIO there are three fixed Device IDs. These are available only when sRIO is active and maybe openly accessed by any of the register reading mechanisms. If sRIO is not active, this section of the die is not powered, and the IDs are not available. The sRIO IDs are as follows:

- The Vendor ID, indicating IDT (assigned by the RapidIO Trade Association)
- The Device ID, indicating the part type
- The die signature, indicating date code, revision or other assembly specific information

JTAG also has a JTAG vendor ID. All JTAG IDs are accessible only through JTAG.

6.2 Memory Map

Base Address	Description
sRIO Configuration Registers	
0x000000 - 0x0000FC	RIO Base Feature Space Registers
0x000100 - 0x00053C	RIO Extended Feature Space Registers
0x000600 - 0x000E3C	RIO Error Management Space Registers
SerB Configuration Registers	
0x018004	Reset & Command Register
0x018008	Serial Port Configuration Register
0x01800C	*reserved for future use*
0x018010	Parallel Port Configuration Register
0x018014	Memory Allocation Register

Table 1 SerB Memory Map

Notes

Base Address	Description
0x018018 - 0x01802C	*reserved for future use*
0x018030	Lost Packet Replacement Register
0x018034	Source and Destination ID Register
0x018038 - 0x018054	*reserved for future use*
0x018058	PAE-PAF Register
0x01805C - 0x018064	*reserved for future use*
0x018068 - 0x018070	Waterlevel/Watermark Control Registers
0x018074 - 0x0180C4	*reserved for future use*
0x0180C8 - 0x0180CC	MBIST Registers
0x0180D0	JTAG Device ID Register
0x0180D4 - 0x0183FC	*reserved*
0x018400 - 0x018410	Case Scenario Configuration Registers
0x018414 - 0x01857C	*reserved for future use*
0x018580 - 0x01858C	Missing Packet Detection Registers
0x018590 - 0x0185BC	*reserved for future use*
0x0185C0 - 0x0185C4	Packet Interval Timer Registers
0x0185C8	*reserved for future use*
0x0185CC	Missing Packet Size Register
0x0185D0 - 0x0185D8	*reserved for future use*
0x0185DC - 0x01860C	S-Port Packet XMT/RCV Counter Registers
0x018610 - 0x018C2C	*reserved*
0x018C30	S-Port SERDES Quad Control Register
0x018C34 - 0x019C00	*reserved*
Flag and Flag Mask Registers	
0x019C04	S-Port Link Status Flag Register
0x019C08	*reserved for future use*
0x019C0C	Device Configuration Error Flag Register
0x019C10	sRIO DMA Status Register
0x019C14 - 0x019C4C	*reserved for future use*
0x019C50	Missing Packet Flag Register
0x019C54 - 0x019C5C	*reserved for future use*
0x019C60	FIFO Queue Empty Flag Register
0x019C64	FIFO Queue Full Flag Register
0x019C68 - 0x019C9C	*reserved for future use*
0x019CA0	DSP Interrupt Flag Register
0x019CA4 - 0x019CC0	*reserved*
0x019CC4	S-Port Link Status Mask Register
0x019CC8	*reserved for future use*
0x019CCC	Device Configuration Error Mask Register
0x019CD0	sRIO DMA Status Mask Register
0x019CD4 - 0x019D0C	*reserved for future use*
0x019D10	Missing Packet Mask Register
0x019D14 - 0x019D1C	*reserved*
0x019D20	FIFO Queue Empty Mask Register
0x019D24	FIFO Queue Full Mask Register

Table 1 SerB Memory Map

Notes

Base Address	Description
0x019D28 - 0x019D5C	*reserved*
0x019D60	Missing Packet Address Log Register 1
0x019D64 - 0x019E0C	*reserved*
0x019E10	Tally Doorbell Flag Register
0x019E14 - 0x019E4C	*reserved*
0x019E50	Missing Packet Programmable Flag Register
0x019E54 - 0x019E5C	*reserved*
0x019E60	DSP Interrupt Mask Register
0x019E64 - 0x019ECC	*reserved*
0x019ED0	Tally Doorbell Mask Register
0x019ED4 - 0x019F0C	*reserved*
0x019F10	Missing Packet Programmable Mask Register
0x019F14 - 0x019F1C	*reserved*
0x019F20	Missing Packet Address Log Registers 2

Table 1 SerB Memory Map

6.3 Configuration Register Programming and Reset

There are multiple types and severity of reset capabilities. Many of the resets involve loading the configuration registers, or clearing values contained in the registers. The various resets may be performed through the following mechanisms:

- External pins
- sRIO control symbols.
- sRIO type 8 maintenance packets
- JTAG and I^2C commands

Multiple types of resets may be generated using the reset mechanisms. The following items list the various resets, the mechanism(s) to force the reset, the effects of the reset and other reset information:

- Master Reset - Performed after power on and anytime a full reset is needed.
  - Pin based reset or sRIO control symbol only.
  - Any shadow registers are programmed to the state required by the hard-wired configuration pins.
  - All configuration registers programmed to the state required by the hard-wired configuration pins.
  - Any registers that do not have default values are cleared.
  - All memory will be cleared.
  - All flag registers will be cleared. All mask registers are set to fully masked.
  - All Error counters and status registers will be cleared (not set to a programmed value).
  - All PLLs will be reset.
  - Any existing state machines will be initialized to a known state.
  - Any changes are immediate
- Partial Reset - Performed anytime and affects all registers. (An example of this type of reset would be the changing of a port data rate).
  - sRIO maintenance packet reset, JTAG, or I^2C based reset. This reset is performed by "hitting" the reset configuration register.
  - Shadow registers are not affected.
  - Configuration registers with shadow registers are programmed to the shadow values.
  - Configuration registers without shadow registers are cleared.
  - All memory will be cleared
  - All flag registers will be cleared. All mask registers are set to fully masked.
  - All Error counters and status registers will be cleared (not set to a programmed value).

Notes

- All PLLs will be reset.
- Any existing state machines will be initialized to a known state.
- Any changes are immediate, except JTAG and I^2C will perform the change at the designated command

- Load Configuration - Identical to "Partial Reset" except ports and PLLs are not reset.

6.3.1 Programming

- Configuration Register Reset (registers without shadows) - These resets may be performed anytime on the fly and they affect only the designated register. They are performed by loading the individual register with a new value. (Example of the registers affected include the destination IDs).
  - sRIO maintenance packet. The registers may not be programmed through JTAG, or I^2C.
  - Shadow registers and configuration registers with shadows are not affected.
  - Memory is not affected
  - Flag registers may be individually cleared using this method. Mask registers are part of the flag register and will be affected along with any writing to the flag registers for clearing.
  - Designated error counters and status registers will be cleared (not set to a programmed value).
  - PLLs are not affected
  - Any existing state machines are not affected, except possibly as a result of the register changing.
  - Any changes are immediate

- Shadow Register Programming - These resets may be performed anytime on the fly and they affect only the designated shadow register. They are performed by loading the individual register with a new value.
  - Programming may be done through sRIO maintenance packets, JTAG or I^2C.
  - Only shadow registers are affected.
  - Memory is not affected
  - Flag and Flag Mask registers are not affected.
  - Error counters and status registers are not affected.
  - PLLs are not affected
  - No existing state machines are affected.
  - There is no immediate effect on any configuration register from programming a shadow register. To load the results of the programming into the designated configuration registers, a "Load Configuration" reset must subsequently be performed.

- Flag Register Reset - These resets may be performed anytime on the fly and they affect only the designated register.
  - Performed with sRIO maintenance packets.
  - Flag registers are cleared by writing to them. Writing the Wr32 bit within the register designates whether the write to a flag register is intended to alter the entire register, including destination IDs, or simply clear flags. Flags may be cleared by writing a "1" to them. Any flag that is written with a "0" will remain unchanged.
  - Some flag registers contain real time values, indicated by "RT" in the flag register section. These values cannot be cleared except by affecting the source of the flag. A new doorbell or interrupt will not be generated if the RT flag is active.
  - Error counters and status registers may be associated with flag registers and will be cleared if written to.
  - JTAG and I^2C may read the registers, but cannot clear the flag registers, except through a load configuration type reset.
  - Any changes are immediate

6.3.2 Clearing Flags

Flags are cleared by the various "resets" associated with the SerB. The methods of clearing flags are described in section 6.3, of this datasheet. In summary, any flag may be cleared by Master Reset, a Load Configuration, or by writing to the flag register. Any mask register may be programmed by writing to it, but it won't be affected by clearing a flag register.

Notes

Any "Real Time" flag, indicated by RT in the following flag register tables, indicates the current condition of the flag-causing event and cannot be cleared. No new doorbells or interrupts will be generated as a result of the write to a flag register containing an RT flag. To generate a new doorbell or interrupt, at least one flag in the register must de-assert and reassert.

Clearable flags are indicated by CL in the following flag register tables. These flags assert and lock whenever a flag event occurs. They must be cleared by one of the designated reset methods. These flags represent highly transient conditions, so in most cases the flag causing condition has disappeared prior to the clearing. In the event that the flag causing condition is active at the time of the clearing, and the flag is immediately reasserted, a new doorbell or interrupt will be generated.

6.3.3 Flag Masks

Flag masks default to fully masked upon a Master Reset or Load Configuration reset. The flag mask registers are considered configuration registers and are individually programmed the same way as other configuration registers. The flag mask registers have no shadow registers, so they can be programmed "on the fly".

Notes

Notes

7.0 Error Management

The SB handles errors in two ways. The errors are defined as either errors that fall under the “RapidIO Part 8: Error Management Extensions Specification” or errors that do not.

The configuration registers associated with errors are found in the RapidIO Part 8: Error Management Extensions Specification section 2.3, outlines the required registers for error management.

This section is focused on errors and status information in addition to the minimum required by the RIO specification.

7.1 sRIO Errors and Error Handling

This section describes how the logical and physical layers will detect and react to RIO errors. The action of the SerB upon notification off any of these errors is described minimally; for detail see Interrupt Generation. Reference RIO Interconnect Specification Part 8 (Error Management Extensions Specifications) for more detail on specific errors described below.

RIO errors are classified under three categories:

- Recoverable errors
- Notification errors
- Fatal errors

7.1.1 Recoverable Errors

These errors are non-fatal transmission errors (such as corrupt packet or control symbols, and general protocol errors) that RIO supports hardware detection of and a recovery mechanism for, as described in the RIO specification. In these cases, the appropriate bit is set in the Port n Error Detect CSR. Only the packet containing the first detected recoverable error that is enabled for error capture (by Port n Error Enable CSR) will be captured in the Port n Error Capture CSRs. No interrupt is generated or actions required for a recoverable error. Recoverable errors are detected in the physical layer only.

7.1.2 Notification Errors

These errors are non-recoverable non-fatal errors detected by RIO (such as Degraded Threshold, Port-Write received, and all logical/transport layer (LTL) errors captured). Because they are non-recoverable (and in some cases have caused a packet to be dropped), notification by interrupt is available. However, because they are non-fatal, response to the interrupt is not crucial to port performance; i.e., the port is still functional. When a notification error is detected, the appropriate bit is set in the error-specific register, an interrupt is generated, and in some cases, the error is captured. The Degraded Threshold error also causes the port to request training (parallel only) with the hope that port performance will improve. In all cases, the RIO port continues operating. Notification errors are detected in both the physical and logical layer.

7.1.3 Fatal Errors

SerB detects two fatal errors:

- Exceeded failed threshold
- Exceeded consecutive retry threshold

In these cases, the port has failed because its Recoverable Error Rate has exceeded a predefined failed threshold or because it has received too many packet retries in a row. In the first case, GRIO will set the Output Failed-encountered bit in the Port n Error and Status CSR; the RIO output hardware may or may not stop (based on Stop-on-Port-Failed-Encounter-Enable and Drop-Packet-Enable bits). In the second case, RIO will set the Retry Counter Threshold Trigger Exceeded bit in the Port n Implementation Error CSR; the RIO hardware will continue to operate. In both cases, an interrupt is generated, and while the port will continue operating at least partially, a system-level fix (such as reset) is recommended to clean up RIO’s internal queues and resume normal operation. Fatal errors are detected in the physical layer only.

7.2 System Software Error Notification

System software is notified of logical, transport, and physical layer errors in two ways. An interrupt is issued to the local system by means of interrupt pins if enabled, or a Maintenance port-write operation issued by SerB. For specifics on interrupt mechanism, see section 5, Doorbells and Interrupt of this datasheet. Maintenance port-write operations are sent to a

Notes

predetermined system host (defined in the Port-write Target deviceID CSR). SerB sets the Port-write Pending status bit in the Port 0 Error and Status CSR. A 16 byte data payload of the Maintenance Port-write packet contains the contents of several CSR, as shown in table below. Once System Software receives an Port-write operation, it indicates that it has seen the port-write by clearing the Port-write Pending status bit in the Port 0 Error and Status CSR.

The Component Tag CSR is defined in the RapidIO Part 3: Common Transport Specification, and is used to uniquely identify the reporting device within the system. A Port ID field contains all 0's indicating port 0, the Logical/Transport Layer Error Detect CSR, and the Port 0 Error Detect CSR are used to describe the encountered error condition.

Data Payload Byte Offset	Word	
0x0	Component TAG CSR	
0x4	Port 0 Error Detect CSR	
0x8	Implementation Specific	PortID(byte)
0xC	Logical/Transport Layer Error Detect CSR	

Table 2 Port-write Packet Data Payload for Error Reporting

7.3 sRIO Errors Supported

7.3.1 Physical Layer Errors

Table below lists all the RIO link errors detected by the SerB physical layer and the actions taken by SerB. The Error Enable column lists the control bits that may disable the error checking associated with a particular error (if blank, error checking cannot be disabled). The Cause Field column indicates what cause field will be used with the associated packet-not-accept control symbol for Input Error Recovery. The EME Error Enable/Detect column indicates which bit of the P0ERECSR allows the error to increment the Error Rate Counter and lock the Port 0 Error Capture registers, and likewise which bit of the P0EDCSR is set when the error has been detected.

Table 4 below, Physical RIO Threshold Response, lists SerB behavior after exceeding certain preset limits (Degraded Threshold, Failed Threshold, Retry Threshold).

Physical RIO Errors Detected

Error	Error Enable	SerB action	Cause Field	EME Error Type	EME Error Enable / Detect
Received character had a disparity error.		Enter Input Error Stopped. Enter Output Error Stopped.	5: Received invalid/illegal character	Delineation Error	DE
Received an invalid character, or valid but illegal character		Enter Input Error Stopped. Enter Output Error Stopped.	5: Received invalid/illegal character		
The four contol character bits associated with the received symbol do not make sense (not 0000, 1000, 1111).		Enter Input Error Stopped. Enter Output Error Stopped.	5: Received invalid/illegal character		
Control symbol does not begin with an /S/ or /PD/ control character.		Enter Input Error Stopped. Enter Output Error Stopped.	5: Received invalid/illegal character		

Table 3 Physical RIO Errors Detected

Notes

Error	Error Enable	SerB action	Cause Field	EME Error Type	EME Error Enable / Detect
Received a control symbol with a bad CRC	P0PCR[CCC] enables detect.	Enter Input Error Stopped. Enter Output Error Stopped.	2. Received a control symbol with bad CRC	Received corrupt control symbol	CCS
Received packet with unexpected ackID (out-of-sequence ACKID).		Enter Input Error Stopped.	1: Received unexpected ACKID on packet	Received packet with unexpected ackID	UA
Received packet with a bad CRC value.	P0PCR[CCP] enables detect.	Enter Input Error Stopped.	4: Bad CRC on packet	Received packet with bad CRC	CRC
Received packet which exceeds the maximum allowed size by the RIO spec.		Enter Input Error Stopped.	7/31: General error	Received packet exceeds 276 Bytes	EM
Packet data received w/o previous SOP control symbol.		Enter Input Error Stopped.	31: General error	Protocol Error (unexpected packet/control symbol received)	PE
Received an EOP control symbol when there is no packet being received.		Enter Input Error Stopped.	7/31: General error		
Received a stomp control symbol when there is no packet being received.		Enter Input Error Stopped.	7/31: General error		
Received packet that is < 64 bits.		Enter Input Error Stopped.	7/31: General error		
Received a Restart-from-retry control symbol when in the “OK” state.		Enter Input Error Stopped.	7/31: General error		
Received packet with embedded idles.		Enter Input Error Stopped.	31: General error		
Received a non-maintenance packet when non-maintenance packet reception is stopped.	Non-maint. packet reception is stopped when “Input Port Enable” = 0.	Enter Input Error Stopped.	3. Non-maintenance packet reception is stopped	Not Captured	
Any packet received while Port Lockout bit is set.	All packet reception is stopped when Port Lockout bit is set.	Enter Input Error Stopped.	3. Non-maintenance packet reception is stopped	Not Captured	
Received a Link request control symbol before servicing previous link request.	Not Detected.				
Received an ACK (accepted or retry) control symbol with an unexpected ACKID.		Enter Output Error Stopped.		Received ack. control symbol with unexpected ackID	AUA
Received packet-not-accepted ACK control symbol		Enter Output Error Stopped.		Received packet-not-accepted symbol	PNA

Table 3 Physical RIO Errors Detected

Notes

Error	Error Enable	SerB action	Cause Field	EME Error Type	EME Error Enable / Detect
Link_response received with an ackID that is not outstanding.		Enter Output Error Stopped.		Non-out-standing ackID	NOA
Received an ACK (accepted, or retry) control symbol when there are no outstanding packets		Enter Output Error Stopped.		Unsolicited ACK symbol	UCS
Received packet ACK (accepted) for a packet whose transmission has nor finish.		Enter Output Error Stopped.			
Received a Link response con-trol symbol when no outstanding request.		Enter Output Error Stopped.			
An ACK control symbol is not received within the specified time-out interval.	PLTOCCSR [TV] > 0 enables detect.	Enter Output Error Stopped.		Link time-out	LTO
A Link response is not received within the specified time-out interval.	PLTOCCSR [TV] > 0 enables detect.	(re-) Enter Output Error Stopped.			

Table 3 Physical RIO Errors Detected

Physical RIO Threshold Response

Error	Error Enable	SerB action	EME Error Type	Error Detect	Interrupt Clear
Notification Errors					
Error Rate Counter has exceeded the Degraded Threshold.	P0ERTCSR[ERDTT]> 0 & any bit in P0EECSR enables detect and interrupt generation.	Generate Interrupt. Parallel port will initiate Maintenance Training if TODTEN bit is set. Con-tinue to operate normally.	Degraded Threshold	P0ESCSR [ODE]	Write 1 to P0ESCSR [ODE]
Fatal Errors					
Consecutive Retry Counter has exceeded the Retry Counter Threshold Trigger.	PRETCR[RET]>0 enables detect and interrupt generation.	Generate Interrupt. Port will be in priority order	Consecutive Retry Threshold	P0IECSR [RETE]	Write 1 to P0IECSR [RETE]
Error Rate Counter has exceeded the Failed Threshold.	P0ERTCSR[ERFTT]> 0 & any bit in P0EECSR enables detect and interrupt generation.	Generate Interrupt. Port behavior depends on P0CCSR[SPF] and P0CCSR[DPE] -- port can continue trans-mitting packets or can stop sending output pack-ets, keeping or dropping them. Parallel port will ini-tiate Full Training if TOFTEN bit is set.	Failed Threshold	P0ESCSR [OFE]	Write 1 to P0ESCSR [OFE]

Table 4 Physical RIO Threshold Response

Notes

7.3.2 Logical Layer Errors

Table below lists all the errors detected by the SerB logical layer and the actions taken by SerB. Note that when the SerB action includes sending an error response to either UL or RIO, an error response is only sent if the original transaction was a request that required a response. Otherwise, no error response is sent. When dealing with multiple errors, discard of packet has higher priority than error response.

Here, error checking is listed based on the type of transaction and table also lists the action taken for particular error.

Errors for NRead Transaction

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Priority of Read transaction is 3	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped

Table 5 Hardware Errors for NRead Transaction

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	Yes	Same as previous entry	--
SourceID Not checked for error					
Transaction-Type Received RIO packet with reserved TType for this ftype	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	
RdSize Not checked for error					
SrcTID Not checked for error					
Address: WdPtr:Xambs Beginning address matches LCSBA1CSR with non 32 bit read request. (Performed only when ttype == 4'b0100)	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--
Header Size Header size is not 12 Bytes for small Trans-port packet or not 16 Bytes for Large Trans-port packet. (Large Trans-port packet has 14 valid bytes and two bytes of 0's. Padding of 0's is not checked).	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--

Table 5 Hardware Errors for NRead Transaction

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
PayloadSize Not Applicable					

Table 5 Hardware Errors for NRead Transaction

Errors for Maintenance Read/Write Request Transaction

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Priority of maintenance read or write request transaction is 3	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped

Table 6 Hardware Errors for Maintenance Read/Write Request Transaction

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	Yes	Same as previous entry	--
SourceID Not checked for error					
Transaction-Type Received RIO packet with reserved TType for this ftype	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--
RdSize Read/Write request size is not for 4 bytes	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--
SrcTID Not checked for error					
HopCount Not checked for error					
Config Offset Not checked for error					

Table 6 Hardware Errors for Maintenance Read/Write Request Transaction

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Header Size Header size is not 12 Bytes for small Transport packet or not 16 Bytes for Large Transport packet. (Large Transport packet has 14 valid bytes and two bytes of 0's. Padding of 0's is not checked).	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--
PayloadSize Write request with payload not equal to 8 bytes Read request with payload not 0 bytes	Bit 4: ITD	Bit 4: ITD	Yes	Same as previous entry	--

Table 6 Hardware Errors for Maintenance Read/Write Request Transaction

Notes

Errors for RIO Write class Transactions

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Nwrite_r, Nwrite transaction has priority 3	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	Yes for Nwrite_r, No for Nwrite	Same as previous entry	--
SourceID Not Applicable					

Table 7 Hardware Errors for RIO Write class Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Transaction-Type Received RIO packet for Atomic test-and-swap transaction Received RIO packet with reserved TType for this ftype Packet is treated as Nwrite Trans-action	Bit 9: UT	Bit 9: UT	Yes	Same as previous entry	--
WrSize WrSize request is for one of reserved sizes	Bit 4: ITD	Bit 4: ITD	Yes for Nwrite_r, No for Nwrite	Same as previous entry	--
SrcTID Not checked for error					
Address: WdPtr:Xambs Nwrite_r address matches LCSBA1CSR with non 32 bit read request. (Performed only when TType == 4'b0101)	Bit 4: ITD	Bit 4: ITD	Yes for Nwrite_r	Same as previous entry	--
Header Size Header size is not 12 Bytes for small Trans-port packet or not 16 Bytes for Large Trans-port packet. (Large Trans-port packet has 14 valid bytes and two bytes of 0's. Padding of 0's is not checked).	Bit 4: ITD	Bit 4: ITD	Yes for Nwrite_r, No for Nwrite	Same as previous entry	--

Table 7 Hardware Errors for RIO Write class Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
PayloadSize Payload is greater than that indicated by {wdptr:wrsize} field, payload is not double word aligned or does not have any payload.	Bit 4: ITD	Bit 4: ITD	Yes for Nwrite_r, No for Nwrite	Same as previous entry	--

Table 7 Hardware Errors for RIO Write class Transactions

Notes

Errors for SWrite class Transactions

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Swrite transaction is priority 3	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	No	Same as previous entry	RIO packet is dropped
SourceID Not Applicable					

Table 8 Hardware Errors for SWrite class Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
PayloadSize Payload size is not in double word aligned, has exceeded 256 bytes or has no payload.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped

Table 8 Hardware Errors for SWrite class Transactions

Errors for Maintenance Response Transactions

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Response priority is not higher than RIO maintenance request priority	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped and ignored

Table 9 Hardware Errors for Maintenance Response Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped and ignored
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	Yes	Same as previous entry	RIO packet is dropped and ignored
SourceID Does not match the request's DestID	Bit 8: UR	Bit 8: UR	No	Same as previous entry	RIO packet is dropped and ignored
Transaction-Type Received RIO packet with reserved TType for this ftype	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
HopCount Not checked for error					
Status Is not "Done" or "Error" Not "Done" status for "read_response" transaction type with payload "Error" status with payload	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
Status Error Response	Bit 0: IER	Bit 0: IER	No	Same as previous entry except error capture is done from original request	--
TargetTID No outstanding transaction for this TargetTID	Bit 8: UR	Bit 8: UR	No	Same as previous entry	RIO packet is dropped and ignored

Table 9 Hardware Errors for Maintenance Response Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Header Size Maintenance Read response - total payload size with done status is not greater than 4 Bytes. Maintenance Write response - total header size is less than 12 Bytes for Small Transport packet or is less than 16 Bytes for Large Transport packet.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
PayloadSize Maintenance write response has payload Maintenance read response with done status and payload not matching valid request size or request size for the response is invalid.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
Packet Response Time-out Response is not received by configured time.	Bit 7: PRT	Bit 7: PRT	No	Same as previous entry except error capture is done from original request	--

Table 9 Hardware Errors for Maintenance Response Transactions

Notes

Error for Response Transaction

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Priority Response priority is not higher than RIO request priority	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped and ignored
TransportType Received reserved TT for this FType Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped and ignored
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	No	Same as previous entry	RIO packet is dropped and ignored

Table 10 Hardware Error for Response Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
SourceID Does not match the request's DestID	Bit 8: UR	Bit 8: UR	No	Same as previous entry	RIO packet is dropped and ignored
Transaction-Type Received RIO packet with reserved TType IO read response does not correspond to an outstand-ing valid IO read request. IO write response does not correspond to an outstand-ing valid IO write request.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
Status IO transaction - Is not "Done" or "Error" Transaction type of "Response_wit h_data" and status is not done.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
Status IO Error Response	Bit 0: IER	Bit 0: IER	Yes	Same as previous entry except error capture is done from original request	--
TargetTID No outstanding transaction for this TargetTID	Bit 8: UR	Bit 8: UR	No	Same as previous entry	RIO packet is dropped and ignored

Table 10 Hardware Error for Response Transactions

Notes

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
Packet Size (All non-maintenance and non-message). Write response - Header size in not 8 Bytes for Small Transport packet or not 12 Bytes for Large Transport packet.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
PayloadSize IO - Read Response - total payload is not of the size requested.	Bit 4: ITD	Bit 4: ITD	No	Same as previous entry	RIO packet is dropped and ignored
Packet response time-out Response is not received by configured time for packets requiring RIO response. Done response is not received in configured time.	Bit 7: PRT	Bit 7: PRT	Yes	Same as previous entry except error capture is done from original request	Interrupt is generated

Table 10 Hardware Error for Response Transactions

Notes

Errors for Reserved FType

Error	Interrupt Generated if enable bit set on LTLEECSR	Status Bit set on LTLEDCSR	RIO Error Response Generated	Logical/Transport Layer Capture Register	Comments
FType Priority of maintenance read or write request transaction is 3	Bit 4: ITD	Bit 4: ITD	No	Using the incoming RIO packet, for Small Transport type packet; LTLACCSR[XA] = packet bits [78:79], LTLACCSR[A] = packet bits [48:76], LTLDIDCCSR[DIDMSB] = 0's, LTLDIDCCSR[DID] = packet bits [16:23], LTLDIDCCSR[SIDMSB] = 0's, LTLDIDCCSR[SID] = packet bits [24:31], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [32:35] For Large Transport type packets; LTLACCSR[XA] = packet bits [94:95], LTLACCSR[A] = packet bits [64:92], LTLDIDCCSR[DIDMSB] = packet bits[16:23], LTLDIDCCSR[DID] = packet bits [24:31], LTLDIDCCSR[SIDMSB] = packet bits[32:39], LTLDIDCCSR[SID] = packet bits [40:47], LTLCCCSR[FT] = packet bits [12:15], LTLCCCSR[TT] = packet bits [48:51]	RIO packet dropped
TransportType Received reserved TT Received TT which is not enabled	Bit 28: TSE	Bit 28: TSE	No	Same as previous entry	RIO packet is dropped
DestID DestID does not match this port's DeviceID or Alternate DeviceID when enabled	Bit 5: ITTE	Bit 5: ITTE	Yes	Same as previous entry	--

Table 11 Hardware Errors for Reserved Ftype

Notes

7.4 Other Serial Buffer Errors

All errors that are not covered by the RapidIO Error Management Extension will be handled by the flag registers and the user programmed reporting methods (flag mask) for those flags. It should be noted that some of the sRIO error are also included in the flag registers and may result in reporting by both the RapidIO Error Management, and the normal flag mask.

Notes

Notes

8.0 Registers

The registers of the SerB are grouped into functions. Register types include the following:

- sRIO Registers (CARs and CSRs)
- SerB Configuration Registers
- SerB Error Counter Registers
- SERDES Control Registers
- Flag & Flag Mask Registers

In the sRIO world, the term CSR is used for “Command and Status Registers”. These are the combination of the configuration and flag registers.

All registers are accessible by S-Port, I^2C and JTAG. Not all parts of the registers are necessarily accessed from all parts. The programming of the configuration registers are described in the section on system initialization. When using sRIO, the configuration registers are accessible only through maintenance packets. They cannot be accessed by using NWRITE, NREAD or SWRITE.

As a further grouping, the electrical characteristics of the ports and presence of external memory should remain fixed once configured, so these should be separated from configurations that may change. It is more likely that destination IDs and other soft configurations will change, especially in large applications that are not adequately served by four output queues on a port.

The configuration registers are broken into blocks of related functions that may be read by any port and written by any port that will not kill itself in process.

It should be noted that in addition to the registers shown here, others exist that are described elsewhere and in the sRIO specification. An example is the Error Management registers that may be found in the RapidIO Part 8: Error Management Extension Specification and in the “Error Handling” section of this document.

8.1 sRIO Registers

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 logical specification. All registers are 32-bits and aligned to a 32-bit boundary.

8.1.1 Register Summary

Table below shows the register map for this RapidIO specification. These capability registers (CARs) and command and status registers (CSRs) can be accessed using RapidIO 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.

8.1.2 Extended Features Data Structure

The RapidIO capability and command and status registers implement an extended capability data structure. If the extended features bit (bit 28) in the processing element features register is set, the extended features pointer is valid and points to the first entry in the extended features data structure. This pointer is an offset into the standard 16 Mbyte capability register (CAR) and command and status register (CSR) space and is accessed with a maintenance read operation in the same way as when accessing CARs and CSRs.

The extended features data structure is a singly linked list of double-word structures. Each of these contains a pointer to the next structure (EF_PTR) and an extended feature type identifier (EF_ID). The end of the list is determined when the next extended feature pointer has a value of logic 0. All pointers and extended features blocks shall index completely into the extended features space of the CSR space, and all shall be aligned to a double-word boundary so the three least significant bits shall equal logic 0. Pointer values not in extended features space or improperly aligned are illegal and shall be treated as the end of the data structure.

Notes

8.1.3 Base Feature Address Space

Block Byte Offset	Register Name (word 0)	Register Name (word 1)
0x00	**Device Identity CAR**	**Device Information CAR**
0x08	**Assembly Identity CAR**	**Assembly Information CAR**
0x10	**Processing Element Features CAR**	Reserved
0x18	**Source Operation CAR**	**Destination Operation CAR**
0x20	Reserved	
0x28	Reserved	
0x30	Reserved (part 11)	Reserved (part 3)
0x38	Reserved (part 11)	Reserved
0x40	Reserved	
0x48	Reserved	**Processing Element Logical Layer CSR**
0x50	Reserved	
0x58	Load Configuration Space Base Address 0 CSR	**Local Configuration Space Base Address 1 CSR**
0x60	**Base Device ID CSR**	Reserved
0x68	**Host Base Device ID Lock CSR**	**Component TAG CSR**
0x70	Reserved (part 3)	Reserved (part 3)
0x78	Reserved (part 3)	Reserved
0x80	Reserved (part 11)	Reserved (part 11)
0x88	Reserved (part 11)	Reserved
0x90 - 0xF8	Reserved	

Table 12 RIO Base Feature Address Space

8.1.4 Capability Registers

The SerB contains a set of Capability Registers (CARs) that allows an external processing element to determine its capabilities through maintenance read operations. All registers are 32 bits wide and are organized and accessed in 32-bit (4 byte) quantities. CARs are read-only and are big-endian with bit 0 the most significant bit.

The use of CARs is described in the RIO Input/Output Logical Specification in Chapter 5.

Device Identity CAR

The Device Identity field identifies the vendor that manufactured the device containing the processing element. A value for the Device Identity field is uniquely assigned to a device vendor by the registration authority of the RIO Trade Association.

The Device Identity field is intended to uniquely identify the type of device from the vendor specified by the Device Identity field. The values for the Device Identity field are assigned and managed by the respective vendor.

Name: **DEV_ID_CAR** **Address:** **0x00000**

Bit	Field Name	Reset Value	Comment
15:0	DEV_VEND_ID	0x0038	Device Vendor Identifier.
31:16	DEV_ID	0x04F0	Device Identifier.

Table 13 Device ID CAR

Notes

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.1

Device Information CAR

The DeviceRev field is intended to identify the revision level of the device. The value for the DeviceRev field is assigned and managed by the vendor specified by the Device Vendor Identity field.

DICAR is a read only register.

Name: **DEV_INFO_CAR** **Address:** **0x00004**

Bit	Field Name	Reset Value	Comment
31:0	DEV_REV	All 0s	Device Revision Level.

Table 14 Device Information CAR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.2

Assembly Identity CAR

The AssyVendorIdentity field identifies the vendor that manufactured the assembly or subsystem containing the device. A value for the AssyVendorIdentity field is uniquely assigned to a assembly vendor by the registration authority of the RIO Trade Association.

The AssyIdentity field is intended to uniquely identify the type of assembly from the vendor specified by the AssyVendorIdentity field. The values for the AssyIdentity field are assigned and managed by the respective vendor.

AIDCAR is a read only register.

Name: **ASSY_ID_CAR** **Address:** **0x00008**

Bit	Field Name	Reset Value	Comment
15:0	ASSY_VEND_ID	0x0000	Assembly Vendor Identifier.
31:16	ASSY_ID	0x0000	Assembly Identifier.

Table 15 Assembly ID CAR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.3

Assembly Information CAR

AICAR contains additional information about the assembly and the pointer to the first entry in the extended features list.

AICAR is a read only register.

Name: **ASSY_INFO_CAR** **Address:** **0x0000C**

Bit	Field Name	Reset Value	Comment
15:0	EXT_FEAT_PTR	0x0100	Extended Features Pointer Field: Pointer to the first entry in the extended features list.
31:16	ASSY_REV	0x0001	Assembly Revision Level.

Table 16 Assembly Info CAR

Notes

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.4

Processing Element Features CAR

PEFCAR identifies the major functionality provided by the processing element.

PEFCAR is a read only register.

Name: **PROC_ELE_FEAT_CAR** **Address:** **0x00010**

Bit	Field Name	Reset Value	Comment
2:0	EXT_ADDR_SUP	3b001	Extended Addressing Support: Indicates the number of address bits supported by the PE both as a source and target of an operation. 3b001 indicates support for 34 bit addresses.
3	EXT_FEAT	1b1	Extended Features: PE has extended features list; the extended features pointer is valid.
4	COM_TRANS_SUP	1b0	Common Transport Large System Support: When enabled it indicates support for 16 bit source and destination ID's.
5	CRF_SUP	1b0	Critical Request Flow Support: 1b0 - PE does not support CRFS 1b1 - PE supports CRFS SerB does not support CRFS, hence this bit is hard wired to zero.
6	RE_TRNS_SUP	1b0	Re-transmit Suppression Support: 1b0 - PE does not support RTSS 1b1 - PE supports RTSS SerB does not support RTSS, hence this bit is hard wired to zero.
7	FLO_CNT_SUP	1b0	Flow Control Support: SerB does not support FCS, hence this bit is hard wired to zero.
8	STD_RTCS	1b0	Standard route table configuration support: SerB does not support SRTCS, hence this bit is hard wired to zero.
9	EXT_RTCS	1b0	Extended route table configuration support: SerB does not support ERTCS, hence this bit is hard wired to zero.
10	MCAST_SUP	1b0	Multicast Extension Support: SerB does not support Multicast, hence this bit is hard wired to zero.
18:11	-	0	Reserved.
19	DOORBELL	1b1	Indicates that the RIO controller supports inbound doorbells.
23:20	MAILBOX	4b0	Mailbox 3:0: Bt 0 indicates PE supports inbound mailbox 0. Bit 1 indicates PE supports inbound mailbox 1. Bit 2 indicates PE supports inbound mailbox 2. Bit 3 indicates PE supports inbound mailbox 3.
27:24	-	0	Reserved.
28	SWITCH	1b0	Indicates that the PE can bridge to another external RIO interface.
29	PROCESSOR	1b0	Indicates that the PE physically contains a local processor that executes code.
30	MEMORY	1b1	Indicates that the PE has physically addressable local address space and can be accessed as an endpoint through non-maintenance operations.
31	BRIDGE	1b0	Indicates that the PE can bridge to another interface.

Table 17 Process Element Features CAR

Notes

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.5, Part 3, sec. 3.4.1, Part 6, sec. 6.4.1, Part 9, sec. 4.2, and Part 11, sec. 3.2

Source Operations CAR

SRCOPCAR defines the set of RIO IO logical operations that can be issued by this processing element.

SRCOPCAR is a read only register.

Name: **SRC_OPS_CAR** **Address:** **0x000018**

Bit	Field Name	Reset Value	Comment
1:0	-	0	Reserved.
2	PORT_WR	1b1	Port Write: PE support a port-write operation.
9:3	-	0	Reserved
10	DBELL	1b1	Doorbell: PE can support a doorbell operation.
11	DATA_MSG	1b0	Data Message: PE can support a data message operation.
12	NWR_W_RESP	1b1	NWRITE_R: PE support a Nwrite_R operation.
13	STRM_WR	1b1	Streaming Write: PE support an Swrite operation.
14	NWRITE	1b1	NWRITE: PE support a Nwrite operation.
15	NREAD	1b1	NREAD: PE support a Nread operation.
31:16	-	0	Reserved.

Table 18 Source Operations CAR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.7, Part 2, sec. 5.4.1, Part 5, sec. 5.4.1, and Part 10, sec. 5.4.1

Destination Operations CAR

DESTOPCAR defines the set of RIO I/O operations that can be supported by this processing element.

DESTOPCAR is a read only register.

Name: **DEST_OPS_CAR** **Address:** **0x00001C**

Bit	Field Name	Reset Value	Comment
1:0	-	0	Reserved.
2	PORT_WR	1b0	Port Write: PE support a port-write operation.
9:3	-	0	Reserved
10	DBELL	1b1	Doorbell: PE can support a doorbell operation.

Table 19 Destination Operations CAR

Notes

Bit	Field Name	Reset Value	Comment
11	DATA_MSG	1b0	Data Message: PE can support a data message operation.
12	NWR_W_RESP	1b1	NWRITE_R: PE support a Nwrite_R operation.
13	STRM_WR	1b1	Streaming Write: PE support an Swrite operation.
14	NWRITE	1b1	NWRITE: PE support a Nwrite operation.
15	NREAD	1b1	NREAD: PE support a Nread operation.
31:16	-	0	Reserved.

Table 19 Destination Operations CAR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.4.8, Part 2, sec. 5.4.2, Part 5, sec. 5.4.2, Part 10, sec. 5.4.2

8.1.5 Command and Status Registers

The SerB contains a set of Command and Status Registers (CSRs) that allows an external processing element to control and determine the 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 5-2 of the RIO Input/Output Logical Specification in Chapter 5 for the required behavior for accesses to reserved registers and register bits.

Processing Element Logical Layer Control CSR

PELLCCSR controls the extended addressing abilities. SerB will only support 34-bit addressing.

PELLCCSR is a read only register.

Name: **PROC_ELMT_CTRL_CSR** **Address:** **0x00004C**

Bit	Field Name	Reset Value	Comment
2:0	EXT_ADDR_CTRL	3b001	Extended Addressing Control (read-only): Controls the number of address bits generated by the PE as a source and processed by the PE as the target of an operation. 3b100 - PE supports 66 bit addresses 3b010 - PE supports 50 bit addresses 3b001 - PE supports 34 bit addresses (default) All other encoding reserved.
31:3	-	0	Reserved.

Table 20 Processing Element Logical Layer Control CSR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.5.1

Local Configuration Space Base Address 1 CSR

The local configuration space base address 1 command and status register specifies the least significant bits of the local physical address double-word offset for the processing element's configuration register space, allowing the configuration register space to be physically mapped in the processing element. This register allows configuration and maintenance

Notes

of a processing element through regular read and write operations rather than maintenance operations. The double-word offset is right-justified in the register. As is the case with all registers, an external processor writing to LCSBA1CSR should not assume it has been written until a response has been received.

Name: **LCL_CONF_ADDR_1_CSR** **Address:** **0x00005C**

Bit	Field Name	Reset Value	Comment
16:0	-	0	Reserved.
30:17	LCL_BASE_ADDR	0x0000	Local Configuration Space Base Address: These bits correspond to the highest 14 bits of the 34-bit RIO address space.
31	-	0	Reserved.

Table 21 Local Configuration Space Base Address 1 CSR

Note:

1. The above register is described in the RIO Specification Part 1, sec. 5.5.3

Base Device ID CSR

The sRIO searchable source and destination IDs are contained in the Base Device ID CSR, and are programmed by sRIO according to the sRIO specification. There are locations for both 8 and 16 bit device IDs as described in the RapidIO, Part 3, Common Transport Specification in section 3.5.1. The SerB shall allow programming of both, in order to allow both 8 and 16 bit operations simultaneously. Both device IDs may be read by any of the interfaces with access to the configuration registers.

The device IDs are cleared only by Master Reset or by a specific write to the Base Device ID CSR. Other resets, such as Load Configuration will have no affect on the Base Device ID CSR. The Base Device ID CSR has no shadow register.

Note: This register is in the sRIO spec and that spec overrides this info.

Name: **BASE_DEV_ID_CSR** **Address:** **0x000060**

Bit	Field Name	Reset Value	Comment
16:0	LRG_BASE_DEVID	0xFFFF	Large Base Device ID: SerB Source/Destination ID is 16 bits. The Base ID of the device in a large common transport system. This field is valid only if bit 27 of the Processing Element Features CAR is set.
30:17	BASE_DEVID	0xFF	Base Device ID: SerB Source/Destination ID is 8 bits. The Base ID of the device in a small common transport system (RIO device ID)
31	-	0	Reserved.

Table 22 Base Device ID CSR

Note:

1. The above register is described in the RIO Specification Part 3, sec. 3.5.1

Host Base Device ID Lock CSR

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 HBDID field is a write-once/resettable field which provides a lock function. Once the HBDID 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 HBDID 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.

Notes

Name: **HOST_BASE_DEV_LOCK_CSR** **Address:** **0x000068**

Bit	Field Name	Reset Value	Comment
15:0	HOST_BASE_DID	0xFFFF	Host Base Device ID: This is the host base device ID for the processing element that is responsible for initializing this device. Only the first write to this field is accepted, all other writes are ignored, except in the case that the value written matches the value contained in the field. In this case, the register is re-written to 0xFFFF.
31:16	-	0	Reserved.

Table 23 Host Base Device ID Lock CSR

Note:

1. The above register is described in the RIO Specification Part 3, sec. 3.5.2

Component Tag CSR

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 unused internally in SerB. It is especially useful for labeling and identifying devices that are not end points and do not have device ID registers.

Name: **COMP_TAG_CSR** **Address:** **0x00006C**

Bit	Field Name	Reset Value	Comment
31:0	COMP_TAG	All 0s	Component Tag: This is a component tag for the PE.

Table 24 Component Tag CSR

Note:

1. The above register is described in the RIO Specification Part 3, sec. 3.5.3

Notes

8.1.6 Extended Features Register Summary

Table below shows the Extended Features register map for this RapidIO specification. These capability registers (CARs) and command and status registers (CSRs) can be accessed using RapidIO maintenance operations. There are four types of 1x/4x LP-Serial devices, as an end point device. SerB supports an end point device with additional software recovery registers.

8.1.7 Extended Features Address Space

Block Byte Offset	Register Name (word 0)	Register Name (word 1)
0x100	**1x/4x LP-Serial Register Block Header**	Reserved
0x108	Reserved	
0x110	Reserved	
0x118	Reserved	
0x120	**Port Link Time-Out Control CSR**	**Port Response Time-Out Control CSR**
0x128	Reserved	
0x130	Reserved	
0x138	Reserved	**Port General Control CSR**
0x140	**Port 0 Link Maintenance Request CSR**	**Port 0 Link Maintenance Response CSR**
0x148	**Port 0 Local ackID Status CSR**	Reserved
0x150	Reserved	
0x158	**Port 0 Error and Status CSR**	**Port 0 Control CSR**
0x160 - 0x178	Reserved for Port 1 Registers	
0x180 - 0x198	Reserved for Port 2 Registers	
0x1A0 - 0x1B8	Reserved for Port 3 Registers	
0x1C0 - 0x538	Reserved for Port 4 through 15 Registers	
0x600	**Error Management Extensions Block Header**	Reserved
0x608	**Logical/Transport Layer Error Detect CSR**	**Logical/Transport Layer Error Enable CSR**
0x610	Logical/Transport Layer High Address Capture CSR	**Logical/Transport Layer Address Capture CSR**
0x618	**Logical/Transport Layer Device ID Capture CSR**	**Logical/Transport Layer Control Capture CSR**
0x620	Reserved	
0x628	Port-write Target deviceID CSR	Packet Time-to-live CSR
0x630 - 0x638	Reserved	
0x640	**Port 0 Error Detect CSR**	**Port 0 Error Rate Enable CSR**
0x648	**Port 0 Attributes Capture CSR**	**Port 0 Packet/Control Symbol Capture 0 CSR**
0x650	**Port 0 Packet Capture 1 CSR**	**Port 0 Packet Capture 2 CSR**
0x658	**Port 0 Packet Capture 3 CSR**	Reserved
0x660	Reserved	
0x668	**Port 0 Error Rate CSR**	**Port 0 Error Rate Threshold CSR**
0x680 - 0x6B8	Reserved for Port 1 Registers	
0x6C0 - 0x6F8	Reserved for Port 2 Registers	
0x700 - 0x738	Reserved for Port 3 Registers	
0x740 - 0xE38	Reserved for Port 4 through 15 Registers	

Table 25 RIO Extended Features Address Space

Notes

1x/4x LP-Serial Register Block Header

The port maintenance block header 0 register contains the EF_PTR to the next EF_BLK (Extended Features Space, Error Management) and the EF_ID that identifies this as the generic end point port maintenance block header. Note that while registers defined by software assisted error recovery are supported, software assisted error recovery is not (these registers are included for hot insertion only); therefore, RIO is defined here as not supporting software assisted error recovery. PMBH0CSR is a read-only register.

Name: **PORT_MAINT_BLK_HDR** **Address:** **0x000100**

Bit	Field Name	Reset Value	Comment
15:0	EF_ID	0x0001	Extended Features ID: Hard wired extended features ID, Generic End Point Devices.
31:16	EF_PTR	0x0600	Extended Features Pointer: Hard wired pointer to the next block in the data structure.

Table 26 1x/4x LP-Serial Register Block Header

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.1

Port Link Time-out Control CSR

The port link time-out control register contains the time-out timer value for all ports on a device. This time-out is for link events such as sending a packet to receiving the corresponding acknowledge and sending a link-request to receiving the corresponding link-response. The reset value is the maximum time-out interval, and represents between 3 and 5 seconds.

Name: **PORT_LNK_TO_CTRL_CSR** **Address:** **0x000120**

Bit	Field Name	Reset Value	Comment
7:0	-	0	Reserved.
31:8	PORT_LINK_VAL	0xFFFFFF	Port Link Time-out Internal Value: Setting to all 0's disables the link time-out timer. This value is loaded each time the link time-out timer starts.

Table 27 Port Link Time-out CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.2

Port Response Time-out Control CSR

The port response time-out control register contains the time-out timer count for all ports on a device. This time-out is for sending a request packet to receiving the corresponding response packet. The reset value is the maximum time-out interval, and represents between 3 and 5 seconds.

Name: **PORT_RESP_TO_CTRL_CSR** **Address:** **0x000124**

Bit	Field Name	Reset Value	Comment
7:0	-	0	Reserved.
31:8	PORT_RESP_VAL	0xFFFFFF	Port Response Time-out Internal Value: Setting to all 0's disables the link time-out timer. This value is loaded each time the link time-out timer starts.

Table 28 Port Response Time-out CSR

Notes

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.3

Port General Control CSR

The port general control register contains control register bits applicable to all ports on a processing element.

Name: **PORT_GEN_CTRL_CSR** **Address:** **0x00013C**

Bit	Field Name	Reset Value	Comment
28:0	-	0	Reserved.
29	DISCOVER	1b0	Discovered: This device has been located by the processing element responsible for system configuration. 0b0 - The device has not been previously discovered. 0b1 - The device has been discovered by another processing element.
30	MSTR_EN	1b0	Master Enable: The master enable bit controls whether or not a device is allowed to issue requests into the system. If the Master Enable is not set, the device may only respond to requests. 0b0 - processing element cannot issue requests. 0b1 - processing element can issue requests.
31	HOST	1b0	Host: A host device is a device that is responsible for system exploration, initialization, and maintenance. Agent or slave devices are typically initialized by Host devices. 0b0 - agent or slave device. 0b1 - host device.

Table 29 Port General Control CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.4

Port 0 Link Maintenance Request CSR

The port 0 link maintenance request register is accessible both by a local processor and an external device. A write to this register generates a link-request control symbol on the corresponding RIO port interface. Care should be taken when writing this register that it is only used for hot swap and not for software assisted error recovery (which is not supported).

Name: **P0_LNK_MAINT_REQ_CSR** **Address:** **0x000140**

Bit	Field Name	Reset Value	Comment
2:0	CMD	3b000	Command: LINK_REQUEST command to send. If read, this field returns the last written value. If written with a value other than 3b011 (reset-device) or 3b100 (input-status), resulting operation will be undefined, as all other values are reserved in the RIO spec.
31:3	-	0	Reserved.

Table 30 Port 0 Link Maintenance Request CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.5

Notes

Port 0 Link Maintenance Response CSR

The port 0 link maintenance response register is accessible both by a local processor and an external device. A read to this register returns the status received in a link-response control symbol. This register is read-only.

Name: **P0_LNK_MAINT_RES_CSR** **Address:** **0x000144**

Bit	Field Name	Reset Value	Comment
4:0	LNKS	0x00	Link Status: link status field from the link-response control symbol.
9:5	ACKS	0x00	ackID Status: ackID status field from the link-response control symbol.
30:10	-	0	Reserved.
31	RVLD	1b0	Response Valid: If the link-request causes a link-response, this bit indicates that the link-response has been received and the status fields are valid. If the link-request does not cause a link-response, this bit indicates that the link-request has been transmitted. This bit automatically clears on read.

Table 31 Port 0 Link Maintenance Response CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.6

Port 0 Local ackID Status CSR

The port n local ackID status register is accessible both by a local processor and an external device. A read to this register returns the local ackID status for both the output and input ports of the device. Care should be taken to use this register only for hot swap and not software error management.

Name: **P0_LOC_ACK_STAT_CSR** **Address:** **0x000148**

Bit	Field Name	Reset Value	Comment
4:0	OBACKID	0x00	Outbound Ack ID: This can be written by software but only if there are no outstanding unacknowledged packets. If there are, a newly-written value will be ignored.
7:5	-	0	Reserved.
12:8	OACKID	0x00	Outstanding port unacknowledge ackID status: Next expected acknowledge control symbol ackID field that indicates the ackID value expected in the next received acknowledge control symbol. Note that this value is read-only even though RIO spec allows for it to be writable.
23:13	-	0	Reserved.
28:24	IACKID	0x00	Inbound ackID: Input port next expected ackID value.
31:29	-	0	Reserved.

Table 32 Port 0 Local ackID Status CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.7

Notes

Port 0 Error and Status CSR

This register is accessed when a local processor or an external device wishes to examine the port error and status information.

Name: **P0_ERR_STAT_CSR** **Address:** **0x000158**

Bit	Field Name	Reset Value	Comment
0	PORT_UNINIT	1b1	Port Uninitialized: Input and output ports are not initialized. This bit and bit 30 are mutually exclusive (read-only).
1	PORT_OK	1b0	Port OK: The input and output ports are initialized and the port is exchanging error-free control symbols with the attached device (read-only).
2	PORT_ERR	1b0	Port Error: Input or output port has encountered an error from which hardware was unable to recover. Once set, remains set until written with a logic 1 to clear.
3	-	0	Reserved.
4	PORT_WR_PEND	1b0	Port-write Pending: Port has encountered a condition which required it to initiate a Maintenance Port-write operation. This bit is only valid if the device is capable of issuing a maintenance port-write transaction. Once set, remains set until written with a logic 1 to clear.
7:5	-	0	Reserved.
8	IN_ERR_STOP	1b0	Input Error-stopped: Input port is stopped due to transmission error (read-only).
9	IN_ERR_ENC	1b0	Input Error-encountered: Input port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 23 is set. Once set, remains set until written with a logic 1 to clear.
10	IN_RTRY_STOP	1b0	Input Retry-stopped: Input port is stopped due to a retry (read-only).
15:11	-	0	Reserved.
16	OUT_ERR_STOP	1b0	Output Error-stopped: Output port is stopped due to a transmission error (read-only).
17	OUT_ERR_ENC	1b0	Output Error-encountered: Output port has encountered (and possibly recovered from) a transmission error. This bit is set when bit 15 is set. Once set, remains set until written with a logic 1 to clear.
18	OUT_RTRY_STOP	1b0	Output Retry-stopped: Output port is stopped due to a retry (read-only).
19	OUT_RETRY	1b0	Output Retried: Output port has received a packet-retry control symbol and can not make forward progress. This bit is set when bit 13 is set and cleared when a packet-accepted or packet-not-accepted control symbol is received (read-only).
20	OUT_RTRY_ENC	1b0	Output Retry-encountered: Output port has encountered a retry condition. This bit is set when bit 13 is set. Once set, remains set until written with a logic 1 to clear.
23:21	-	0	Reserved.

Table 33 Port 0 Error and Status CSR

Notes

Bit	Field Name	Reset Value	Comment
24	OUT_DGRD_ENC	1b0	Output Degraded-encountered: Output port has encountered a degraded condition, meaning that the Error Rate Counter has met or exceeded the port's degraded error threshold. Once set remains set until written with a logic 1 to clear. Once cleared, will not assert again unless the Error Rate Counter dips below the port's degraded error threshold and then meets or exceeds it again.
25	OUT_FAIL_ENC	1b0	Output Failed-encountered: Output port has encountered a failed condition, meaning that the Error Rate Counter has met or exceeded the port's failed error threshold. Once set, remains set until written with a logic 1 to clear. Once cleared, will not assert again unless the Error Rate Counter dips below the port's failed error threshold and then meets or exceeds it again.
26	OUT_PKT_DROP	1b0	Output Packet-dropped: Output port has discarded a packet. A packet will be discarded if: 1. it is received while OFE is set and drop packet enable is set and stop on port failed is set. 2. it is received while output buffer drain enable is set. 2. it is not-accepted by the link-partner while error rate failed threshold trigger is met or exceeded and link-response returns expected ackID. Once set, it remains set until written with a logic 1 to clear.
31:27	-	0	Reserved.

Table 33 Port 0 Error and Status CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.8

Port 0 Control CSR

The port 0 control register contains control register bits for the individual port on a processing element.

Name: **P0_CTRL_CSR** **Address:** **0x000158**

Bit	Field Name	Reset Value	Comment
0	PORT_TYPE	1b0	Port Type, this indicates the port type (read-only): 1b0 - Port receiver/drivers are enabled 1b1 - Port receivers/drivers are disabled and are unable to receive/ transmit any packets or control symbols
1	PORT_LOCK	1b0	Port Lockout: 1b0 - The packets that may be received and issued are controlled by the state of the OPE and IPE bits. 1b1 - This port is stopped and is not enabled to issue or receive any packets.
2	DROP_PKT_EN	1b0	Drop Packet Enable: This bit is used with the Stop on Port Failed-encountered Enable bit to force certain behavior when the Error Rate Failed Threshold has been met or exceeded.
3	STOP_PORT_FAIL	1b0	Stop on Port Failed-encountered Enable: This bit is used with the Drop Packet Enable bit to force certain behavior when the Error Rate Threshold has been met or excessed.

Table 34 Port 0 Control CSR

Notes

Bit	Field Name	Reset Value	Comment
11:4	RE_XMT_MASK	0x00	Re-transmit Suppression Mask: Suppress packet re-transmission on CRC error. SerB does not support this feature and these bits are set to zero.
16:12	-	0	Reserved.
17	ENUM_BOUN	1b0	Enumeration Boundary: An enumeration boundary aware system enumeration algorithm shall honor this flag. The algorithm, on either the ingress or the egress port, shall not enumerate past a port with this bit set. This provides for software enforced enumeration domains within the RIO fabric.
18	FLO_CTRL_PART	1b0	Flow Control Participant, enable flow control transactions: 1b0 - Do not route or issue flow control transactions to this port 1b1 - Route or issue flow control transactions to this port. (RIO spec. Part 9, sec. 4.3)
19	MULTI_PART	1b0	Multicast-event Participant: This bit is hard-wired to 0.
20	ERR_CHK_DIS	1b0	Error Checking Disable, this bit disables all RIO transmission error checking: 1b0 - error checking and recovery is enabled 1b1 - error checking and recovery is disabled
21	IN_PORT_EN	1b0	Input Port Enable, input port receive enable: 0b0 - port is stopped and only enabled to route or respond to I/O logical MAINTENANCE packets. 0b1 - port is enabled to respond to any packet.
22	OUT_PORT_EN	1b0	Output Port Enable, output port transmit enable: 1b0 - port is stopped and not enabled to issue any packets except to route or respond to I/O Logical Maintenance packets. 1b1 - port is enabled to issue any packets.
23	PORT_DIS	1b0	Port Disable: 1b0 - Port receiver/drivers are enabled 1b1 - Port receivers/drivers are disabled and are unable to receive/transmit any packets or control symbols
26:24	PORT_OVER	3b000	Port Width Override, soft port configuration to override the hardware size: 3b000 No override 3b001 Reserved 3b010 Force single lane, lane 0 3b011 Force single lane, lane 2 3b100 - 3b111 Reserved The change of this field during normal mode may cause re-initialization.
29:27	INIT_PORT_WDTH	HW	Initialized Port Width, width of the ports after initialized (read-only): 3b000 Single-lane port, lane 0 3b001 Single-lane port, lane 2 3b010 Four-lane port 3b011 - 3b111 Reserved.
31:30	PORT_WIDTH	HW	Port Width, hardware width of the port (read-only): 2b00 Single-lane port 2b01 Four-lane port 2b10 - 2b11 Reserved.

Table 34 Port 0 Control CSR

Note:

1. The above register is described in the RIO Specification Part 6, sec. 6.6.2.9

Notes

8.1.8 Error Management Extensions Summary

Error Management Extensions Block Header

The error management extensions block header register contains the EF_PTR to the next EF_BLK and the EF_ID that identifies this as the error management extensions block header.

Name: **ERR_MGMT_BLK_HDR** **Address:** **0x000600**

Bit	Field Name	Reset Value	Comment
15:0	EXT_FEAT_ID	0x0007	Extended Features ID: Hard wired extended features ID.
31:16	EXT_FEAT_PTR	0x0000	Extended Features Pointer: Hard wired pointer to the next block in the data structure.

Table 35 Error Management Extensions Block Header

Note:

The above register is described in the RIO Specification Part 8, sec. 2.3.2.1

Logical/Transport Layer Error Detect CSR

This register indicates the error detected by the Logical or Transport logic layer. Multiple bits may get set in the register if simultaneous errors are detected during the same clock cycle that the errors are logged, or if the detected errors are not enabled for capture. LTLEDCSR is stored in each GRIO port and the Message Unit, although the values in this register can differ for each port/Message Unit. A port's LTLEDCSR cannot detect any errors if the port or the Message Unit has captured an enabled Logical/Transport layer error until the detected error is cleared, and likewise, the Message Unit's LTLEDCSR cannot detect any errors if the Message Unit or any port has captured an enabled Logical/Transport layer error. Software should write this register with all 0's to clear the detected error and unlock the capture registers in all ports/ Message Unit. Undefined results will occur if this register is written or read while actual Logical/Transport Layer errors are being detected by the port (where detect cannot occur if an error has already been detected and not yet cleared).

If a port detects multiple errors in the same cycle, multiple LTLEDCSR bits will be set to reflect this. If one or all of these bits are enabled, capture is done on a priority basis. If PRT is set and enabled, and multiple bits are detected in LTLEDCSR, the capture information corresponds to PRT. If PRT is not set or not enabled, then all set and enabled LTLEDCSR bits correspond to the captured packet.

If more than one port or Message Unit detects one or more enabled errors in the same cycle, the capture registers will be saved in the top port /Message Unit in the PBUS daisy chain that detected an enabled error, and the set and enabled detect bits of the port(s)/Message Unit below will be masked from the PBUS daisy chain. This means that a read of LTLEDCSR will only return the un-enabled set bits from any port/Message Unit and enabled set bits from the top port / Message Unit in the daisy chain with a set enabled error, and that a read of the capture registers will return the values in the top port /Message Unit in the daisy chain with a set enabled error; i.e., the set enabled detect bits will correspond to the capture registers.

Name: **LTL_ERR_DET_CSR** **Address:** **0x000608**

Bit	Field Name	Reset Value	Comment
2:0	-	0	Reserved.
3	TRSP_SZE_ERR	1b0	Transport Size Error: The tt field is nor consistent with bit 27 of the Processing Element Features CAR (i.e., the tt value is reserved or indicates a common transport system that is unsupported by this device).

Table 36 Logical/Transport Layer Error Detect CSR

Notes

Bit	Field Name	Reset Value	Comment
4	RTRY_TRES_EXC	1b0	Retry Error Threshold Exceeded: The allowed number of logical retries has been exceeded.
21:5	-	0	Reserved.
22	UNSUP_TRANS	1b0	Unsupported Transaction: A transaction is received that is not supported in the Destination Operation CAR (IO/MSG/GSM logical).
23	UNSOL_RES	1b0	Unsolicited Response: An unsolicited/unexpected Response packet was received (IO/MSG/GSM logical).
24	PKT_RES_TOUT	1b0	Packet Response Time-out: A required response has not been received within the specified time-out interval (IO/MSG/GSM logical).
25	MSG_REQ_TOUT	1b0	Message Request Time-out: A required message request has not been received within the specified time-out interval (MSG logical).
26	ILL_TRANS_ERR	1b0	Illegal Transaction Target Error: Received a packet that contained a destination ID that is not defined for this end point.
27	ILL_TRANS_DEC	1b0	Illegal Transaction Decode: Received illegal fields in the request/response packet for a supported transaction (IO/MSG/GSM logical).
28	MSG_FMT_ERR	1b0	Message Format Error: Received MESSAGE packet data payload with an invalid size or segment (MSG logical).
29	GSM_ERR_RES	1b0	GSM Error Response: Received a response of 'ERROR' for a GSM Logical Layer Request.
30	MSG_ERR_RES	1b0	Message Error Response: Received a response of 'ERROR' for an MSG Logical Layer Request.
31	IO_ERR_RES	1b0	IO Error Response: Received a response of 'ERROR' for an IO Logical Layer Request.

Table 36 Logical/Transport Layer Error Detect CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.2

Logical/Transport Layer Error Enable CSR

This register contains the bits that control if an error condition locks the Logical/Transport Layer Error Detect and Capture registers and is reported to the system host. LTLEECSR is stored in all ports and the Message Unit

Name: **LTL_ERR_EN_CSR** **Address:** **0x00060C**

Bit	Field Name	Reset Value	Comment
2:0	-	0	Reserved.
3	TRAN_SZE_EN	1b0	Transport Size Error Enable: Enable error reporting when the tt field is nor consistent with bit 27 of the Processing Element Features CAR (i.e., the tt value is reserved or indicates a common transport system that is unsupported by this device).

Table 37 Logical/Transport Layer Error Enable CSR

Notes

Bit	Field Name	Reset Value	Comment
4	RE_TRS_EXC_EN	1b0	Retry Error Threshold Exceeded Enable: Enable error reporting when all allowed number of logical retries has been exceeded.
21:5	-	0	Reserved.
22	UNS_TRANS_EN	1b0	Unsupported Transaction Error Enable: Enable reporting of an unsupported transaction error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.
23	UNS_RES_EN	1b0	Unsolicited Response Error Enable: Enable reporting of an unsolicited response error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.
24	PKT_RES_TO_EN	1b0	Packet Response Time-out Error Enable: Enable reporting of a packet response time-out error. Save and lock original request address in Logical/Transport Layer Address Capture CSRs. Save and lock original request Destination ID in Logical/ Transport Layer Device ID Capture CSRs.
25	MSG_REQ_TO_EN	1b0	Message Request Time-out Enable: Enable reporting of a Message Request time-out error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs for the last Message request segment packet received.
26	ILL_TRGT_EN	1b0	Illegal Transaction Target Error Enable: Enable reporting of an illegal transaction target error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.
27	ILL_DEC_EN	1b0	Illegal Transaction Decode Enable: Enable reporting of an illegal transaction decode error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.
28	MSG_FRMT_EN	1b0	Message Format Error Enable: Enable reporting of a message format error. Save and lock transaction capture information in Logical/Transport Layer Device ID and Control Capture CSRs.
29	GSM_ERR_EN	1b0	GSM Error Response Enable: Enable reporting of a GSM error response. Save and lock original request transaction information in all Logical/Transport Layer Capture CSRs.
30	MSG_ERR_EN	1b0	Message Error Response Enable: Enable reporting of a Message error response. Save and lock original request transaction information in all Logical/Transport Layer Capture CSRs.
31	IO_ERR_EN	1b0	IO Error Response Enable: Enable reporting of an IO error response. Save and lock original request transaction information in all Logical/Transport Layer Capture CSRs.

Table 37 Logical/Transport Layer Error Enable CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.3

Notes

Logical/Transport Layer Address Capture CSR

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set. LTLACCSR is stored in each port and the Message Unit, although the values in this register can differ between each port and Message Unit. The Message Unit LTLACCSR cannot lock if any port has locked; no port LTLACCSR can lock if the Message Unit or any other port has locked. Undefined results will occur if this register is written while actual Logical/Transport Layer errors are being detected by the port.

Name: **LTL_ADDR_CAP_CSR** **Address:** **0x000614**

Bit	Field Name	Reset Value	Comment
1:0	EXTA	2b00	xamsbs: Extended address bits of the address associated with the error (for requests, for responses if available).
2	-	0	Reserved.
31:3	ADDR	All 0s	address[32:60]: Least significant 29 bits of the address associated with the error (for requests, for responses if available).

Table 38 Logical/Transport Layer Address Capture CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.5

Logical/Transport Layer Device ID Capture CSR

This register contains error information. It is locked when a Logical/Transport error is detected and the corresponding enable bit is set. LTLDIDCSR is stored in each port and the Message Unit, although the values in this register can differ between each port and Message Unit. The Message Unit LTLDIDCSR cannot lock if any port has locked; no port LTLDIDCSR can lock if the Message Unit or any other port has locked. Undefined results will occur if this register is written while actual Logical/Transport Layer errors are being detected by the port.

Name: **LTL_DEV_ID_CSR** **Address:** **0x000618**

Bit	Field Name	Reset Value	Comment
7:0	SRC_ID	0x00	Source ID: The sourceID (or least significant byte of the source ID if large transport system) associated with the error.
15:8	MSB_SRC_ID	0x00	MSB Source ID: The most significant byte of the sourceID associated with the error. This field is valid only if bit 27 of the Processing Element Features CAR is set (large transport systems only).
23:16	DST_ID	0x00	Destination ID: The destinationID (or least significant byte of the destination ID if large transport system) associated with the error.
31:24	MSB_DST_ID	0x00	MSB Destination ID: The most significant byte of the destinationID associated with the error. This field is valid only if bit 27 of the Processing Element Features CAR is set (large transport systems only).

Table 39 Logical/Transport Layer Device ID Capture CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.6

Notes

Logical/Transport Layer Control Capture CSR

This register contains error information. LTLCCCSR is stored in each port and the Message Unit, although the values in this register can differ between each port and Message Unit. The Message Unit LTLCCCSR cannot lock if any port has locked; no port LTLCCCSR can lock if the Message Unit or any other port has locked. Undefined results will occur if this register is written while actual Logical/Transport Layer errors are being detected by the port.

Name: **LTL_CTRL_CAP_CSR** **Address:** **0x00061C**

Bit	Field Name	Reset Value	Comment
15:0	-	0	Reserved.
23:16	MSG_INFO	0x00	Message Information: Letter, mbox, and message for the last Message request received for the mailbox that had an error (Message errors only).
27:24	TRANS_TYPE	0x0	Transaction Type: Transaction type associated with the error.
31:28	FORMAT_TYPE	0x0	Format Type: Format type associated with the error.

Table 40 Logical/Transport Layer Control Capture CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.7

Port-write Target deviceID CSR

This register contains the target device ID to be used when a device generates a Maintenance port-write operation to report errors to a system host.

Name: **PORT_WR_TID_CSR** **Address:** **0x000628**

Bit	Field Name	Reset Value	Comment
14:0	-	0	Reserved.
15	LRG_TRANS	1b0	Large Transport: DeviceID size to use for a port-write 1b0 - use the small transport deviceID 1b1 - use the large transport deviceID.
23:16	DEV_ID	0x00	DeviceID: This is the port-write target deviceID.
31:24	DEV_ID_MSB	0x00	DeviceID MSB: This is the most significant byte of the port-write target deviceID (large transport systems only).

Table 41 Port-write Target deviceID CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.8

Port 0 Error Detect CSR

The Port 0 Error Detect Register indicates transmission errors that are detected by the hardware. Software can write bits in this register with “1” to cause the Error Rate Counter to increment. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port.

Notes

Name: P0_ERR_DET_CSR **Address:** 0x000640

Bit	Field Name	Reset Value	Comment
0	LINK_TOUT	1b0	Link Time-out: An acknowledge or link-response control symbol is not received within the specified time-out interval.
1	UNS_CTRL_SYM	1b0	Unsolicited Acknowledge Control Symbol: An unexpected acknowledge control symbol was received.
2	DELIN_ERR	1b0	Delineation Error: Received unaligned /SC/ or /PD/ or undefined code-group.
3	-	0	Reserved.
4	PROTO_ERR	1b0	Protocol Error: An unexpected packet or control symbol was received.
5	NOUT_ACKID	1b0	Non-outstanding ackID: Link-response received with an ackID that is not outstanding.
16:6	-	0	Reserved.
17	RCV_PKT_EXC	1b0	Received Packet Exceeds 276 Bytes: Received packet which exceeds the maximum allowed size.
18	RCV_BAD_CRC	1b0	Received Packet with bad CRC: Received packet with a bad CRC value.
19	RCV_PKT_UACK	1b0	Received Packet with Unexpected ackID: Received packet with unexpected ackID value (out-of-sequence ackID).
20	RCV_PKT_NCTRL	1b0	Received Packet-not-accepted Control Symbol: Received packet-not-accepted acknowledge control symbol.
21	RCV_ACK_SYM	1b0	Received Acknowledge Control Symbol with Unexpected ackID: Received acknowledge control symbol with unexpected ackID (packet-accepted or packet-retry).
22	RCV_CC_SYM	1b0	Received Corrupt Control Symbol: Received a control symbol with a bad CRC value.
31:23	-	0	Reserved.

Table 42 Port 0 Error Detect CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.10

Port 0 Error Rate Enable CSR

This register contains the bits that control when an error condition is allowed to increment the error rate counter in the Port 0 Error Rate Threshold Register and lock the Port 0 Error Capture registers.

Name: P0_ERR_RATE_EN_CSR **Address:** 0x000644

Bit	Field Name	Reset Value	Comment
0	LINK_TOUT_EN	1b0	Link Time-out Enable: Enable error rate counting of link time-out errors.

Table 43 Port 0 Error Rate Enable CSR

Notes

Bit	Field Name	Reset Value	Comment
1	UNS_ACK_SYM_EN	1b0	Unsolicited Acknowledge Control Symbol Enable: Enable error rate counting of unsolicited acknowledge control symbol errors.
2	DELIN_ERR_EN	1b0	Delineation Error Enable: Enable error rate counting of delineation errors.
3	-	0	Reserved.
4	PROTO_ERR_EN	1b0	Protocol Error Enable: Enable error rate counting of protocol errors.
5	NOUT_ACKID_EN	1b0	Non-outstanding ackID Enable: Enable error rate counting of link-response received with an ackID that is not outstanding.
16:6	-	0	Reserved.
17	RCV_PKT_EXC_EN	1b0	Received Packet Exceeds 276 Bytes: Enable error rate counting of packet which exceeds the maximum allowed size.
18	RCV_BAD_CRC_EN	1b0	Received Packet with bad CRC Enable: Enable error rate counting of packet with a bad CRC value.
19	RCV_PKT_ACK_EN	1b0	Received Packet with Unexpected ackID Enable: Enable error rate counting of packet with unexpected ackID value (out-of-sequence ackID).
20	RCV_PKT_SYM_EN	1b0	Received Packet-not-accepted Control Symbol Enable: Enable error rate counting of received packet-not-accepted control symbols.
21	RCV_ACK_SYM_EN	1b0	Received Acknowledge Control Symbol with Unexpected ackID Enable: Enable error rate counting of an acknowledge control symbol with an unexpected ackID.
22	RCV_CC_SYM_EN	1b0	Received Corrupt Control Symbol Enable: Enable error rate counting of a corrupt control symbol.
31:23	-	0	Reserved.

Table 43 Port 0 Error Rate Enable CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.11

Port 0 Attribute Capture CSR

The error capture attribute register indicates the type of information contained in the port n error capture registers. In the case of multiple detected errors during the same clock cycle one of the errors must be reflected in the Error type field. The error that is reflected is implementation dependent. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port. Also, there could be latency between asserting an interrupt from Output-Degraded Encountered or Output-Failed Encountered to loading this register, such that the interrupt is asserted a few cycles before the error is captured into this register.

Notes

Name: **P0_ATTR_CAP_CSR** **Address:** **0x000648**

Bit	Field Name	Reset Value	Comment
0	CAP_VALID_INFO	1b0	Capture Valid Info: This bit is set by hardware to indicate that the Packet/control symbol capture registers contain valid information. For control symbols, only capture register 0 will contain meaningful information.
7:1	-	0	Reserved.
23:8	EXT_CAPT_INFO	0x0000	Extended Capture Information[0:15]: ECI contains the control/data character signal corresponding to each byte of captured data. ECI[0] = bit associated with P0PSC0CSR[0:7] ECI[1] = bit associated with P0PSC0CSR[8:15] ECI[2] = bit associated with P0PSC0CSR[16:23] ECI[3] = bit associated with P0PSC0CSR[24:31] ECI[4] = bit associated with P0PSC1CSR[0:7] ECI[5] = bit associated with P0PSC1CSR[8:15] ... ECI[14] = bit associated with P0PSC3CSR[16:23] ECI[15] = bit associated with P0PSC3CSR[24:31]
28:24	ERR_TYPE	0x0	Error Type: The encoded value of the bit in the Port 0 Error Detect CSR that describes the error captured in the Port 0 Error Capture CSRs.
29	-	0	Reserved.
31:30	INFO_TYPE	2b00	Info Type, type of information logged: 2b00 - packet 2b01 - control symbol (only error capture register 0 is valid) 2b10 - implementation specific 2b11 - undefined.

Table 44 Port 0 Attribute Capture CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.12

Port 0 Packet/Control Symbol Capture 0 CSR

This register contains the first 4 bytes of captured packet symbol information or a control character and control symbol. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port. Also, there could be latency between asserting an interrupt from Output-Degraded Encountered or Output-Failed Encountered to loading this register, such that the interrupt is asserted a few cycles before the error is captured into this register.

Name: **P0_PKT_CAP_0_CSR** **Address:** **0x00064C**

Bit	Field Name	Reset Value	Comment
31:0	CAPT_0	All 0s	Capture 0: Control character and control symbol or Bytes 0 to 3 of Packet Header.

Table 45 Port 0 Packet/Control Symbol Capture 0 CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.13

Notes

Port 0 Packet Capture 1 CSR

Error capture register 1 contains bytes 4 through 7 of the packet header. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port. Also, there could be latency between asserting an interrupt from Output-Degraded Encountered or Output-Failed Encountered to loading this register, such that the interrupt is asserted a few cycles before the error is captured into this register.

Name: **P0_PKT_CAP_1_CSR** **Address:** **0x000650**

Bit	Field Name	Reset Value	Comment
31:0	CAPT_1	All 0s	Capture 1: Control character and control symbol or Bytes 4 to 7 of Packet Header.

Table 46 Port 0 Packet/Control Symbol Capture 1 CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.14

Port 0 Packet Capture 2 CSR

Error capture register 2 contains bytes 8 through 11 of the packet header. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port. Also, there could be latency between asserting an interrupt from Output-Degraded Encountered or Output-Failed Encountered to loading this register, such that the interrupt is asserted a few cycles before the error is captured into this register

Name: **P0_PKT_CAP_2_CSR** **Address:** **0x000654**

Bit	Field Name	Reset Value	Comment
31:0	CAPT_2	All 0s	Capture 2: Control character and control symbol or Bytes 8 to 11 of Packet Header.

Table 47 Port 0 Packet/Control Symbol Capture 2 CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.15

Port 0 Packet Capture 3 CSR

Error capture register 3 contains bytes 12 through 15 of the packet header. Undefined results will occur if this register is written while actual physical layer errors are being detected by the port. Also, there could be latency between asserting an interrupt from Output- Degraded Encountered or Output-Failed Encountered to loading this register, such that the interrupt is asserted a few cycles before the error is captured into this register.

Name: **P0_PKT_CAP_3_CSR** **Address:** **0x000658**

Bit	Field Name	Reset Value	Comment
31:0	CAPT_3	All 0s	Capture 3: Control character and control symbol or Bytes 12 to 15 of Packet Header.

Table 48 Port 0 Packet/Control Symbol Capture 3 CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.16

Notes

Port 0 Error Rate CSR

The Port 0 Error Rate register is a 32-bit register used with the Port 0 Error Rate Threshold register to monitor and control the reporting of transmission errors.

Name: **P0_ERR_RATE_CSR** **Address:** **0x000668**

Bit	Field Name	Reset Value	Comment
7:0	ERR_RATE_CNTR	0x00	Error Rate Counter: These bits maintain a count of the number of transmission errors that have been detected by the port, decremented by the Error Rate Bias mechanism, to create an indication of the link error rate. Software should not attempt to write this field to a value higher than failed threshold trigger plus the number of errors specified in the ERR field (the maximum ERC value).
15:8	PEAK_ERR_RATE	0x00	Peak Error Rate: This field contains the peak value attained by the error rate counter.
17:16	ERR_RATE_REC	2b00	Error Rate Recovery: These bits limit the incrementing of the error rate counter above the failed threshold trigger: 2b00 - only count 2 errors above 2b01 - only count 4 errors above 2b10 - only count 16 errors above 2b11 - do not limit incrementing the error rate count Note that the Error Rate Counter will never increment above 0cFF, even if the combination of the settings of ERR and the failed threshold trigger might imply that it would.
23:18	-	0	Reserved.
31:24	ERR_RATE_BIAS	0x80	Error Rate Bias: These bits provide the error rate bias value: 0x00 - do not decrement the error rate counter 0x01 - decrement every 1ms (+/-34%) 0x02 - decrement every 10ms (+/-34%) 0x04 - decrement every 100ms (+/-34%) 0x08 - decrement every 1s (+/-34%) 0x10 - decrement every 10s (+/-34%) 0x20 - decrement every 100s (+/-34%) 0x40 - decrement every 1000s (+/-34%) 0x80 - decrement every 10000s (+/-34%) Other values are reserved and will cause undefined operation.

Table 49 Port 0 Error Rate CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.17

Port 0 Error Rate Threshold CSR

The Port 0 Error Rate Threshold register is a 32-bit register used to control the reporting of the link status to the system host.

Notes

Name: **P0_ERR_RATE_CSR** **Address:** **0x00066C**

Bit	Field Name	Reset Value	Comment
15:0	-	0	Reserved.
23:16	ERR_DEG_TRIG	0xFF	Error Rate Degraded Threshold Trigger: These bits provide the threshold value for reporting an error condition due to a degrading link. 0x00 - Disable the Error Rate Degraded Threshold Trigger 0x01 - Set the error reporting threshold to 1 0x02 - Set the error reporting threshold to 2 ... 0xFF - Set the error reporting threshold to 255.
31:24	ERR_FAIL_TRIG	0xFF	Error Rate Failed Threshold Trigger: These bits provide the threshold value for reporting an error condition due to a possibly broken link: 0x00 - Disable the Error Rate Failed Threshold Trigger 0x01 - Set the error reporting threshold to 1 0x02 - Set the error reporting threshold to 2 ... 0xFF - Set the error reporting threshold to 255.

Table 50 Port 0 Error Rate Threshold CSR

Note:

1. The above register is described in the RIO Specification Part 8, sec. 2.3.2.18

8.2 Configuration Registers

The configuration registers are grouped into functions with a maximum of 32 bits per register. Every configuration register is assigned a reference number for ease of location. The reference number may be used as a pointer to the address of the register whenever the configuration address is being loaded or read. The registers are read or programmed as described in the programming section 6 of this datasheet. The registers are shown with bit 0 assigned as the LSB of the register and bit 31 assigned as the MSB. Flag registers are 64 bits long with bit 63 assigned as the MSB.

Within the configuration registers there are five types of bits. The bit type is shown in the column labeled “type”. The five types are:

HW Hard wired bits that are set by the hard wired configuration of the device. These cannot be changed by any programming method and are not affected by any of the resets. These bits may be read by any of the designated methods for reading configuration registers. These primarily deal with port structure and electrical connections. The shadow is the external pin.

RST Bits that will enter a default mode based upon the hard-wired configuration during Master Reset. Subsequently these bits may be changed by any of the designated programming methods, and then performing a “load configuration” reset. These primarily deal with internal device structure. These registers must have a shadow register. Whenever a register containing RST bits are read by any of the designated reading methods, the actual content of the register is returned and not the content of the shadow register.

RW Bits that may be changed at any time without a Load Configuration reset. In the event JTAG or I^2C is used to alter the content, a Load Configuration Reset must be performed. These primarily deal with data routing and flagging. These registers have no shadow, except for the JTAG and I^2C registers. These bits may be read by any of the designated methods.

RO Read Only. Upon a master reset or load configuration, these will go to a known state, but once initialized they are under control of the SerB internally. sRIO Transaction IDs are an example of a register that the

Notes

interface must increment with each transaction, the user may read the register, but the user cannot change the transaction IDs without causing a sequence error.

RC Read to Clear. These bits are associated with MBIST. Contained within the MBIST register is a bit to indicate BIST is done. These bits will clear on read, only if MBIST is complete.

In addition, the configuration registers have a default mode. The defaults are shown in the column labeled “reset value”. The reset values have the following form:

HW The bit is set by the hard-wired pin configuration during reset. Whether the user can subsequently change or not depends upon the type. Protocols, port usage, etc. may affect the status of this bit.

0 Bit defaults to zero

1 Bit defaults to one

X Bit must be programmed before use. The initial state is not a concern.

8.2.1 Reset and Command Register

This register may be written in order to perform a reset and other functions. The bits automatically clear after performing the function, allowing the user to write again to perform an additional reset without having to clear the bits. The use of any of these bits will clear the memory and reset all state machines. The bits are listed in priority, with Master Reset overriding Partial Reset and Partial Reset overriding Load Configuration.

Name: **RST_CMD_REG** **Address:** **0x18004**

Bits	Field Name	Type	Reset Value	Comment
0	MR_RST	RW	1b0	Master reset: Hard Reset. The device will default to the hard wired configuration
1	PR_RST	RW	1b0	Partial Reset: Loads the shadow into the configuration registers and resets PLLs
2	LD_CFG	RW	1b0	Load Configuration: Loads the shadow into the configuration registers
31:3	-		0	Reserved

Table 51 Reset and Command Register

Note:

1. See Section 6.3 for a complete description and functionality of these resets.
2. Partial reset must be used if the port configuration is changed.
3. Does not reset PLLs and cannot be used if the port configuration was changed.
4. There is a master reset used by sRIO described in the RapidIO Part 4: Physical Layer 8/16 LP VLDS Specification.

8.2.2 Serial Port Configuration Register

The ***Serial Port Configuration Register*** sets the speed of S-Port. The serial port configuration register will default to the configuration designated on the hard-wired inputs upon Master Reset. Once set, the register may be reconfigured as described. At any time, full read access is available from all indicated ports.

Notes

Name: SPORT_CFG_REG **Address:** 0x18008

Bits	Field Name	Type	Reset Value	Comment
1:0	SP_SPEED	RST	HW	S-Port Speed Select: 00 = 1.25G, 01 = 2.5G, 10 = 3.125G, 11 = reserved
11:2	-		0	Reserved
12	816_RIO_DESTID	RST	HW	8 or 16 bit sRIO Destination ID: 0 = 8-bit destination ID, 1 = 16-bit destination ID
31:13	-		0	Reserved

Table 52 Serial Port Configuration Register

8.2.3 Parallel Port Configuration Register

The ***Parallel Port Configuration Register*** is used to set up the external QDRII SRAM memory.

The default configuration is dependent upon the hard-wired inputs. A change to the contents of this register will also clear all memory contents in the queue. The registers may be read at any time, without upsetting the content.

Name: PPORT_CFG_REG **Address:** 0x18010

Bit	Field Name	Type	Reset Value	Comment
0	PPORT_ON	RST	HW	Parallel Port On/Off: 0 = On, 1 = Off
1	-		0	Reserved for future use
3:2	EXT_MEM_SZ	HW	HW	External Memory Size: 00 = 36M, 01 = 72M, 1X = reserved
31:4	-		0	Reserved

Table 53 P-Port Configuration Register

Note:

8.2.4 Memory Allocation Register

The ***Memory Allocation Register*** is used to allocate both the internal and external memory to queue 0. Internal memory is available for allocation in block size that are 1/8th of the total SRAM capacity, hence 8 register bits[15:8]. The external memory allocation is dependent upon the size of the external memory attached, but is available in block sizes that are 1/4th of the total external memory capacity and hence 4 register bits[7:4].

Name: MEM_ALLOC_REG **Address:** 0x18014

Bit	Field Name	Type	Reset Value	Comment
3:0	-		4h0	Reserved
7:4	EXT_MEM_BLK	RW	4h0	External Memory Block Allocation
15:8	INT_MEM_BLK	RW	8hFF	Internal Memory Block Allocation
31:16	-		0	Reserved

Table 54 Memory Allocation Register

Note:

Notes

8.2.5 Lost Packet Replacement Register

Name: LOST_PKT_REP_REG **Address:** 0x18030

Bit	Field Name	Type	Reset Value	Comment
0	REP_LOST_PKT	RW	1b0	Queue 0 Replace Lost Packets: 0 = No, 1 = Yes
31:1	-		0	Reserved

Table 55 Lost Packet Replacement Register

Note:

0) If a single packet is lost, it will be replaced by a dummy packet to avoid breaking the memory addresses. If more than one packet is lost, an error will be generated instead of replacing. If this bit is set to 0, the lost packet is ignored.

8.2.6 Source and Destination IDs

The sRIO source and destination IDs must be programmed to access the queue. These registers are read/write from any of the access ports. sRIO may program these registers using the overall device destination ID, since direct sRIO access to the queue may otherwise not be available. In addition, a queue input or output may be programmed with either an 8 or 16 bit destination ID.

Name: SRC_DEST_ID_REG **Address:** 0x18034

Bit	Field Name	Type	Reset Value	Comment
7:0	SRC_ID_8	RW	8h0	Source ID is 8 bits: Defaults to base queue number
15:8	SRC_ID_16	RW	8h0	Source ID is 16 bits: Defaults to zero
23:16	DEST_ID_8	RW	8h0	Destination ID is 8 bits: Defaults to base queue number
31:24	DEST_ID_16	RW	8h0	Destination ID is 16 bits: Defaults to zero

Table 56 Source and Destination ID Register

Note:

7:0) This is the queue source ID for sRIO protocol.

15:8) This is the 16 bit extension for sRIO. These bits will be compared if the queue input is enabled for 16 bits.

23:16) This is the queue destination ID for sRIO.

31:24) This is the 16 bit extension for sRIO on the destination ID. These bits will be appended to the sRIO header if the queue output is enabled for 16 bits.

8.2.7 Program Almost Empty / Almost Full Register

The PAE and PAF flags are each eight bits long for each of the queues. The eight bits allow the flags to be placed anywhere with an accuracy of $1/256^{th}$ of the total queue size. Since the queue size is programmable, PAF and PAE are proportional indications and not accurate size counts. The PAE flag is the distance from empty and the PAF flag is the distance from Full. Both may be placed anywhere in the memory, but cannot overlap to where PAF + PAE > 0FFh.

These registers may be read or written from any of the sources.

Notes

Name: PAE_PAF_REG **Address:** 0x18058

Bit	Field Name	Type	Reset Value	Comment
7:0	PAE_Q0	RW	8h0F	Program Almost Empty
15:8	PAF_Q0	RW	8h0F	Program Almost Full
31:16	-		16h0F0F	Reserved

Table 57 PAE / PAF Register

8.2.8 Waterlevel Control Registers

There is a waterlevel associated with each queue. If the waterlevel is used, the watermark should be set to zero. If the queue is a doorbell master, the watermark should be set at maximum.

Watermark Register

Name: WATER_MARK_ REG **Address:** 0x18068

Bit	Field Name	Type	Reset Value	Comment
22:0	WATER_MARK	RW	23h0	Watermark: Waterlevel trigger point
23	DW_PKT_CNT	RW	1b0	D-Word or Packet Count: 0 = Count Packets, 1 = Count D-Words
31:24	-		0	Reserved

Table 58 Watermark Register

Waterlevel Register

Name: WATER_LEVEL_ REG **Address:** 0x1806C

Bit	Field Name	Type	Reset Value	Comment
22:0	WATER_LEVEL	RO	23h0	Waterlevel: Quantity in queue, in D-Words or packets
31:23	-		0	Reserved

Table 59 Waterlevel Register

Space Available Register

Name: SPACE_ AVAIL_REG **Address:** 0x18070

Bit	Field Name	Type	Reset Value	Comment
22:0	SPC_AVAIL	RO	23h0	Space Available: Remaining space in D-Words, will update to available space on next few clock cycles
31:23	-		0	Reserved

Table 60 Space Available Register

Notes

Note:

1. The watermark is the trigger point at which the flag will be set. As a master that will always transmit new data as soon as it has arrived and been accepted, the watermark should be set to zero.
2. D-Word or Packet count indicates whether the watermark and waterlevel are in terms of packet count or in D-Word count.
3. Flush or Single Packet determines what happens when data is sent out of the queue.
   a. On flush, all data in the queue is transmitted, except for new data that arrives during the flush.
   b. On Single Packet, only enough data is sent to lower the waterlevel below the watermark. Presumably, in most situations, this will be a single packet or D-Word.
   c. It should be noted that the Flush or Single Packet works with the Master/Slave selection in the *Serial Port Configuration Register*. If the queue is a master, the waterlevel triggers the data transmission. If a slave, the waterlevel triggers a flag only and the queue may then be read.

8.2.9 MBIST Control Register

The MBIST is the primary method for memory testing. The MBIST register is one of the few configuration registers with clear on read on most bits. It is expected that all BIST will be controlled by one location/Port, preventing conflicts that may develop from interacting ports, making the clear on read a valid operational mode.

Name: **CONFIG_REG_MBIST** **Address:** **0x180C8**

Bit	Field Name	Type	Reset Value	Comment
0	MBIST_START	RW	1b0	Memory BIST Start: This bit self clears after MBIST is complete
1	MBIST_EN	RW	1b0	Memory BIST Enable: This bit is read/write, must stay high during MBIST
2	I2C_MEM_EN	RW	1b0	I^2C Memory Access Enable: Bits 1 and 2 are XOR
7:3	-		0	Reserved
15:8	MBIST_MEM_ERR	RT	8h0	Memory BIST Main Memory Block Error: Block 7 - 0
20:16	-		0	Reserved
21	MB_P1_SR_ME	RT	1b0	Memory BIST Port 1 / sRIO Memory Error
22	MB_P2_PP_ME	RT	1b0	Memory BIST Port 2 / Parallel Port Memory Error
23	-		0	Reserved
24	MB_DONE	RT	1b0	Memory BIST Done: If this bit is not “1”, the flags from 8 -25 will not clear on read
25	MB_PASS	RT	1b1	Memory BIST Pass This bit is meaningful only when bit 24 = 1
31:26	-		0	Reserved

Table 61 MBIST Control Register

Note:

1. MBIST will start when bit 1 is 1, and bit 0 changes from 0 to 1. Bit 1 will stay at "1" till MBIST is done (bit 24 becomes 1), after that, bit 1 will be self cleared to 0.

8.2.10 QBIST Control Register

The QBIST accompanies the MBIST register. Most bits are clear on read.

Notes

Name: CONFIG_REG_QBIST **Address:** 0x180CC

Bit	Field Name	Type	Reset Value	Comment
0	-		0	Reserved
1	QBIST_EN	RW	1b0	QBIST Enable; This bit is R/W and must stay high during QBIST. Changing from 0 to 1 will reset bits 23:8 and 25.
2	I2C_MEM_EN	RW	1b0	I^2C Memory Access Enable; Bits 1 and 2 are XOR
7:3	-		0	Reserved
23:8	QBIST_ERR_CNT	RC	16h0	QBIST Error Counter Block 7 - 0
24	-		0	Reserved
25	QBIST_PASS	RC	1b1	QDR Memory BIST Pass
31:26	-		0	Reserved

Table 62 QBIST Control Register

Note:

QBIST will start once QBIST enable changes from 0 to 1 and stop when QBIST enable changes from1 to 0.

2:1) Bits 1 and 2 are exclusive of each other.

23:8) The error counter will wrap around once saturated. The user may check this counter against a timer to determine the bit error rate.

8.2.11 JTAG Device ID Register

JTAG Device Identification register is provided for use with identifying the device. The content is duplicated here to allow access from all available access ports.

Name: JTAG_DEVICE_ID **Address:** 0x180D0

Bit	Field Name	Type	Reset Value	Comment
0	STDRD_BIT	HW	0b1	Standard Bit: Standard bit[0] = 1 per IEEE-2001.
11:1	IDT_JTAG_ID	HW	0x033	IDT JTAG Identification: JTAG Vendor ID for IDT.
27:12	IDT_PART_NUM	HW	0x04F0	IDT JTAG Part Number: JTAG Device ID for SerB. 0x4F0 - sRIO / 18Meg 0x4F1 - sRIO / 9Meg
31:28	IDT_VER_NUM	HW	0x0	IDT Version Number: Version number of SerB = 0.

Table 63 JTAG Device ID Register

8.2.12 Case Scenario Configuration Registers

Case scenarios are used to generate sRIO outgoing packet headers when the SerB initiates a packet. In the case of response packets, the incoming packet is used instead. A complete description is provided in the Case Scenario section.

These registers are read/write from any of the access ports. The default values are functionally don't care, since they cannot be used until programmed.

Notes

Case Scenario Packet Header Register

Name: **CS0_PKT_HEADER** **Address:** **0x18400**

Bit	Field Name	Type	Reset Value	Comment
1:0	PRIORITY	RW	2b0	sRIO Priority Packet
3:2	TT	RW	2b0	Transaction Type, 00 = 8 bit, 01 = 16 bit
7:4	FTYPE	RW	4h0	sRIO Transaction Format Type
15:8	TARGETADDR	RW	4h0	Destination ID for the transmission
23:16	TARGETADDR16	RW	8h0	Extension for 16 bit if TT = 01; see note 1
27:24	TTYPE	RW	4h0	Transaction Type (sub group of FTYPE)
31:28	-		0	Reserved

Table 64 Case Scenario Packet Header Register

Note:

1:0) PRIORITY - The priority for the sRIO packet header. CR is set to zero and ignored as part of the priority. Default priority should be 00h, low priority.

3:2) TT - The sRIO transaction type. If set to 00h, the transaction is 8 bits, if set to 01h, the transaction is 16 bits. Other TT values are invalid.

7:4) FTYPE - Defined in the sRIO specification, part 1, section 4.1. The only FTYPEs supported are types 5 (WRITE) and 6 (SWRITE).

15:8) TARGET ADDRESS - The destination ID for the packet to be sent. This byte will be included in all packets using this case scenario.

23:16) TARGET ADDRESS, x16 - The MSB of the address if the sRIO transaction is 16 bits. If TT = 00, the target address MSBs are used.

27:24) TTYPE - The sub transaction to the FTYPE defined in the same location as FTYPE. If FTYPE is 5 the only TTYPEs supported are NWRITE and NWRITE_R.

SIZE: The size is set by the hardware and should not be part of the case scenario. See sRIO spec., section 4.1.2.

Case Scenario Start Address Register

The starting address for memory writes when performing SWRITE and NWRITE operations with this case scenario. The address contained in the packet will increment appropriately starting from this location. Upon a wrap or reset, the address will return to this value.

Name: **CS0_STRT_ADDR** **Address:** **0x18404**

Bit	Field Name	Type	Reset Value	Comment
30:0	STRT_ADDR	RW	31h0	Start Address: Starting memory address for sRIO
31	-		0	Reserved

Table 65 Case Scenario Start Address Register

Case Scenario Next Address Register

Notes

The current value for the ADDRESS. Whenever a new case scenario is programmed, this value will be set to be identical to the START ADDRESS. The address will increment by the quantity of data transmitted with every packet. The NEXT ADDRESS will not rise beyond the STOP ADDRESS. If the Wrap or Stop bit in the following register is set to WRAP, the NEXT address will reset to the START ADDRESS whenever STOP ADDRESS has been hit. If the wrap occurs in the middle of the packet, the NEXT ADDRESS will increment after the reset to indicate how much of the tail of the packet was written after the wrap.

Name: **CS0_NEXT_ADDR** **Address:** **0x18408**

Bit	Field Name	Type	Reset Value	Comment
30:0	NEXT_ADDR	RW	31h0	Next Address: Current memory address for sRIO
31	-		0	Reserved

Table 66 Case Scenario Next Address Register

Case Scenario Stop Address Register

The final incremental address for writing. The higher address may be included in the sRIO packet header, but in some cases the packet length may cause a write to an address higher than this value, in case of an overflow. The START ADDRESS and STOP ADDRESS should be identical if the user does not want the address issued in the packet header to increment.

Name: **CS0_STOP_ADDR** **Address:** **0x1840C**

Bit	Field Name	Type	Reset Value	Comment
30:0	STOP_ADDR	RW	31h0	Stop Address: Maximum memory address for sRIO
31	-		0	Reserved

Table 67 Case Scenario Stop Address Register

Case Scenario Frame Register

Name: **CS0_FRAME_REG** **Address:** **0x18410**

Bit	Field Name	Type	Reset Value	Comment
9:0	FRAME_SIZE	RW	10h0	Frame size
14:10	FRAME_OFFSET	RW	5h0	Frame Offset: Offset on first count
15	TALLY_FLAG	RW	1b0	Set Tally Flag: 1 = Send doorbell to Dest ID on count = size
23:16	FRAME_COUNT	RW	8h0	Frame Count: Counts frame, reset on doorbell
27:24	-		0	Reserved
28	MEM_WRAP_STP	RW	1b0	Memory Wrap or Stop: Stop or Wrap on STOP Address

Table 68 Case Scenario Frame Register

Notes

Bit	Field Name	Type	Reset Value	Comment
29	FLAG_WRAP_STP	RW	1b0	Set Flag on Wrap or Stop: Used with either stop or wrap
31:30	FRAME_CNT	RO	2b0	The highest two bits of the frame count

Table 68 Case Scenario Frame Register

Note:

9:0) FRAME SIZE - The maximum frame size of the data for the TI application. Whenever the FRAME SIZE has been hit, a doorbell will be issued to wake up the DSP (if doorbell is enabled).

14:10) FRAME OFFSET - TI requested that we have an offset to the first FRAME SIZE to allow them to compensate for delays through the system.

15) DOORBELL - This bit indicates that the FRAME SIZE is active and the doorbell should be sent when FRAME SIZE is hit.

23:16) FRAME COUNT - The location for the current value of the counter for FRAME SIZE. When the count reaches FRAME SIZE and the doorbell is active, the doorbell will be sent and FRAME COUNT will reset to zero.

27:24) Lite Dest ID - Lite protocols have only four bits to select either a destination ID or Case Scenario. To solve the problem of what happens when a Lite protocol selects a case scenario and then the packet needs to be loaded into a queue, the Lite Dest ID is placed in the case scenario. The queue inputs may be programmed to allow selection of multiple queues with the same destination ID.

28) Memory Wrap or Stop - Defines whether the NEXT ADDRESS will wrap or stop when it hits STOP ADDRESS. 0 = WRAP, 1 = STOP.

29) Memory Doorbell - Indicates whether a doorbell should be sent when the NEXT ADDRESS hits the STOP ADDRESS.

31:30) Frame size plus offset should not exceed ten bits.

8.2.13 Missing Packet Detection Registers

Missing Packet Detection mechanism consists of Memory Start Address, Current Memory Address, Memory Address Increment and Memory Stop Address registers and are fully described in the section on Missing Packet Detection and Replacement.

Memory Start Address Register

Name: **MEM_STRT_ADDR** **Address:** **0x18580**

Bit	Field Name	Type	Reset Value	Comment
30:0	MEM_STRT_ADDR	RW	31h0	Memory Start Address: Start address for missing packet detection
31	-		0	Reserved

Table 69 Missing Packet Start Address Register

Current Memory Address Register

Notes

Name: CNT_MEM_ADDR **Address:** 0x18584

Bit	Field Name	Type	Reset Value	Comment
30:0	CNT_MEM_ADDR	RW	31h0	Current Memory Address: Used to hold the current memory address
31	-		0	Reserved

Table 70 Missing Packet Current Address Register

Memory Address Increment Register

Name: MEM_ADDR_MEM **Address:** 0x18588

Bit	Field Name	Type	Reset Value	Comment
5:0	MEM_ADDR_INC	RW	6h0	Memory Address Increment: Used to predict next current memory address
31:6	-		0	Reserved

Table 71 Missing Packet Address Increment Register

Memory Stop Address Register

Name: MEM_STOP_ADDR **Address:** 0x1858C

Bit	Field Name	Type	Reset Value	Comment
30:0	MEM_STOP_ADDR	RW	31h0	Memory Stop Address: The last allowed memory address
31	-		0	Reserved

Table 72 Missing Packet Stop Address Register

Note:

1. The stop address must align with the start address and the address increment. A misalignment may cause the stop address to be missed.

8.2.14 Packet Interval Timer Register

The PPS has no storage capability and cannot accept packets faster that its processing capability. To solve this problem, both the data packets and the doorbells exiting S-Port 1 may be timed with a programmable interval timer. The interval timer uses the PHY clock 156.25MHz as the tick. A set interval is programmed into the PPS Packet Interval Timer Register. When a packet is sent using sRIO out S-Port 1, the interval timer will begin counting down, starting when the packet has completed. When the counter reaches zero, a following packet may be sent. The PPS acceptance of doorbells is much faster than data packets: therefore, they will be accomplished by a second counter with the countdown initiated when the doorbell starts.

Data Packet Interval Timer Register

Notes

Name: DATA_ ITV_TIME **Address:** 0x185C0

Bit	Field Name	Type	Reset Value	Comment
15:0	DATA_PKT_TIME	RW	16h0	Data Packet Timer: Counts down, holds at 00h
31:16	-		0	Reserved

Table 73 Data Packet Interval Timer Register

Doorbell Packet Interval Timer Register

Name: DB_ITV_ TIME **Address:** 0x185C4

Bit	Field Name	Type	Reset Value	Comment
15:0	DBELL_PKT_TIME	RW	16h0	Doorbell Packet Timer: Counts down, holds at 00h
31:16	-		0	Reserved

Table 74 Doorbell Packet Interval Timer Register

8.2.15 Missing Packet Size Register

This register is used to set the size of missing/replacement packet payload.

Name: MISS_PKT_SZ **Address:** 0x185CC

Bit	Field Name	Type	Reset Value	Comment
5:0	MISS_PKT_SZ	RW	6h0	Missing Packet Size: Size for inserted packet payload
31:6	-		0	Reserved

Table 75 Missing Packet Size Registers

8.2.16 Missing Packet Address Logging Register 1

Upon the detection of a missing packet, the address of the next valid packet will be loaded into this register. The user may then poll this register to identify the address.

Name: MISS_PKT_LOG_1 **Address:** 0x19D60

Bit	Field Name	Type	Reset Value	Comment
30:0	MIS_PKT_LOG_1	Note	31h0	Missing Packet Address Log: The address of the first valid packet following a missing packet
31	-		0	Reserved

Table 76 Missing Packet Address Logging Register

Note:

30:0) The address of the next valid packet following a missing packet is loaded into this register whenever the missing packet flag is toggled. When either the Missing Packet Flag register is read and cleared, this register will also clear.

Notes

8.2.17 Missing Packet Address Logging Register 2

This register is identical to the ***Missing Packet Address Logging Register***, except it is associated with the "***Missing Packet Programmable Flag Register***" instead of the "***Missing Packet Flag Register***".

Name: **MISS_PKT_LOG_2** **Address:** **0x19F20**

Bit	Field Name	Type	Reset Value	Comment
30:0	MIS_PKT_LOG_2	Note	31h0	Missing Packet Address Log for TI DSP: The address of the first valid packet following a missing packet
31	-		0	Reserved

Table 77 Missing Packet Address Logging Register for TI DSP

Note:

30:0) The address of the next valid packet following a missing packet is loaded into this register whenever the missing packet flag is toggled. When either the Missing Packet Flag register is read and cleared, this register will also clear.

8.3 SerB Error Counter Registers

8.3.1 S-Port Data Packet Received Counter

As part of the device error management, there is a data packet received counter associated with S-Port. This counter is reset by reading. The counter will count every data packet entering the port. Upon reaching full count, the packets will remain at full count and will not wrap.

Name: **DATA_PKT_RCV_CNT** **Address:** **0x185DC**

Bit	Field Name	Type	Reset Value	Comment
31:0	SDP_RX_CNT	RW	32h0	S-Port Data Packet Received Counter: Reset 0 by reading

Table 78 S-Port Data Packet Received Counter

8.3.2 S-Port Data Packet Transmitted Counter

As part of the device error management, there is a data packet transmitted counter associated with S-Port. This counter is reset by reading. The counter will count every data packet leaving the port. Upon reaching full count, the packets will remain at full count and will not wrap.

Name: **DATA_PKT_XMT_CNT** **Address:** **0x185E0**

Bit	Field Name	Type	Reset Value	Comment
31:0	SDP_TX_CNT	RW	32h0	S-Port Data Packet Transmitted Counter: Reset 0 by reading

Table 79 S-Port Data Packet Transmitted Counter

8.3.3 S-Port Priority Packet Received Counter

As part of the device error management, there is a priority packet received counter associated with S-Port. This counter is reset by reading. The counter will count every priority packet entering the port. Upon reaching full count, the packets will remain at full count and will not wrap.

Notes

Name: **PRIO_PKT_RCV_CNT** **Address:** **0x185E4**

Bit	Field Name	Type	Reset Value	Comment
31:0	SPP_RX_CNT	RW	32h0	S-Port Priority Packet Received Counter: Reset 0 by reading

Table 80 S-Port Priority Packet Received Counter

Note:

1. Single 32-bit aggregate counter to count all the non-blocking (Config Write Request and Config Read Request) and blocking (sRIO NREAD and Doorbell Request Frame and Lite Read) priority packets being received on Port 1 Interface.

8.3.4 S-Port Priority Packet Transmitted Counter

As part of the device error management, there is a priority packet transmitted counter associated with S-Port. This counter is reset by reading. The counter will count every priority packet leaving the port. Upon reaching full count, the packets will remain at full count and will not wrap.

Name: **PRIO_PKT_XMT_CNT** **Address:** **0x185E8**

Bit	Field Name	Type	Reset Value	Comment
31:0	SPP_TX_CNT	RW	32h0	S-Port Priority Packet Transmitted Counter: Reset 0 by reading

Table 81 S-Port Priority Packet Transmitted Counter

Note:

1. Single 32-bit aggregate counter to count all the priority packets (Doorbell Request, NWRITE Response, Config Read Response, Config Write Response and Doorbell Response) being transmitted on Port 1 Interface.

8.3.5 S-Port Packet Received Counters

As part of the device error management, there is a packet received counter associated with the queue. These counters are reset by reading. Each counter will count every packet entering the queue. Upon reaching full count, the packets will remain at full count and will not wrap.

Name: **SP_PKT_RCV_CNT** **Address:** **0x185EC**

Bit	Field Name	Type	Reset Value	Comment
31:0	SPKT_RCV_CNT	RW	32h0	S-Port Packet Received Counter: Reset 0 by reading

Table 82 S-Port Packet Received Counter

8.3.6 S-Port Packet Transmitted Counters

As part of the device error management, there is a packet transmitted counter associated with the queue. These counters are reset by reading. Each counter will count every packet sent from the queue. Upon reaching full count, the packets will remain at full count and will not wrap.

Notes

Name: SP_PKT_XMT_CNT **Address:** 0x1860C

Bit	Field Name	Type	Reset Value	Comment
31:0	SPKT_XMT_CNT	RW	32h0	S-Port Packet Transmitted Counter: Reset 0 by reading

Table 83 S-Port Packet Transmitted Counter

8.4 SERDES Quad Control Register

The sRIO specification has defined registers for use in configuring and controlling the 1x/4x Quad Serdes sRIO port (S-Port 1 on the SerB). The SerB shall utilize the standard register and observe standard 1x/4x configuration protocols.

For the rest of the serial ports definition, refer to “RapidIO Interconnect Specification Part VI: Physical Layer 1x/4x LP-Serial Specification.

Name: SERDES_QUAD_CTRL **Address:** 0x18C30

Bit	Field Name	Type	Reset Value	Comment
1:0	-		0	Reserved
4:2	TCOEFF[2:0]	RW	3b0	Transmit pre-emphasis control: 000 = 0% emphasis 001 = 6.5% emphasis 010 = 13% emphasis 011 = 19.5% emphasis 100 = 26% emphasis 101 = 32.5% emphasis 110 = 39% emphasis 111 = 45.5% emphasis
6:5	-		0	Reserved
9:7	TXDRVSEL	RW	3b010	Tx drive strength select 000 = maximum drive strength 010 = sRIO long haul 100 = sRIO short haul 111 = minimum drive strength
31:10	-		0	Reserved

Table 84 SERDES Quad Control Register

8.5 Flag and Flag Mask Registers

The flag registers are 32-bit registers and include an additional 32-bit register for the flag masks. Each register contains a maximum of 8 flags plus the masks and destination IDs associated with those flags. The typical flag register content is shown below table. The flags within a register are selected to generate same interrupt or generate doorbells destined for the same location. The interrupting flag may individually be identified by the register contents that may be read or sent with a doorbell.

Contained within each flag register is a series of four mask registers for the flags. The flag mask registers are used to create doorbells and interrupts. This means there are five register locations associated with each flag.

The content of each flag register is available for reading at any time by any of the following methods:

- sRIO commands
- I^2C Interface
- JTAG

Notes

#	Signal	Stat	Description
7-0	FLAGS	X	There are up to 8 flags contained in the register
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	RES	X	Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	RES	X	Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 85 Flag and Flag Mask Register

Note:

7:0) These 8 bits within the register are where the flags are stored. As noted below, there is a name, a status for the flag and a description of the flag shown for each flag. Some flags are updated real time, while others latch and must be cleared.

8) Designates whether source and destination IDs of the doorbell should use 8 bits or 16 bits. The true TT is two bits, but only one bit is required to make 8/16 designation.

9) Wr32 designates whether a write to this register is 8 bits or 32 bits. An 8 bit write would write a mask to the flag portion of the register to clear the masked flags. Within the mask, any flag that is overwritten by "1" would be cleared. Any flag overwritten with a "0" would be unaffected by the write. A 32 bit write to this register would be required to alter the destination ID and TT portions of the register. In a 32 bit write, the flags could be cleared or left unaffected, based upon the state of the 8 LSBs of the write, the same as with the 8 bit write. Note that RT flags cannot be cleared.

11:10) These bits indicate the priority that should be used for any sRIO doorbell packet.

15:12) Any unused bits are indicated as Reserved.

23:16) This is the 8 bit destination ID for a doorbell on S-Port if the mask bits 8-15 allow a doorbell to be created.

31:24) This is the 16 bit extension to the destination ID for doorbells on S-Port if using sRIO extended addresses.

39:32) These are the mask bits for the flags. Any unmasked flag will cause a doorbell to be sent on S-Port.

47:40) Reserved for future use.

55:48) This is the mask for the Int 0 interrupt pin.

63:56) This is the mask for the Int 1 interrupt pin.

8.5.1 Key to the Flag Registers

The flag registers are listed in the order of priority for doorbells and interrupts. All masks power up and reset to fully masked and are enabled by unmasking the bits.

All flag register tables are shown with the following column headings and symbols within the register:

- #: The bit location within the designated register
- Signal: An abbreviated name for the Flag
- Stat: Indicates whether the flag may be cleared
  - CL: Clearable flag. These flags will latch upon toggling and must be cleared by a write to the register
  - RT: Indicates the flag is a real time and always represents current conditions. Clearing a RT flag is not possible
- RW: Used with masks to indicate the bits are Read/Write through a configuration read/write

Notes

8.5.2 sRIO Link Status

The RapidIO “Error Management Extensions Specification” requires specific Configuration Status Registers at designated addresses. These CSRs are described in section 2 of the identified spec, and should be referenced for more specific information.

8.5.3 S-Port Link Status

The flags of the “S-Port Link Status”. This flag register is used to identify error that is not covered by the sRIO Error Management Extensions Specification.

Name: **SP_LNK_STAT_FLAG** **Address:** **0x19C04**
SP_LNK_STAT_MASK **0x19CC4**

This register cannot generate an sRIO packet.

#	Signal	Stat	Description
3:0	-		Reserved
4	RETRY	CL	sRIO Doorbell Response with RETRY received
5	JTAG	CL	JTAG error
6	EME	CL	sRIO Error Management Extension Interrupt
7	-		Unused bit
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 86 S-Port 1 Link Status for Lite Register

Note:

4) Indicates that in response to a previously sent doorbell a “RETRY” indication was received.

8.5.4 Device Configuration Error

These flags are generated whenever a configuration error occurs. When a configuration error occurs the SerB will not function, however these flags may be masked to create doorbells and interrupts on the ports designated by the masks. All of these flags are real time (RT) and cannot be cleared except by re-configuring the offending register.

Name: **CONFIG_ERR_FLAG** **Address:** **0x19C0C**
CONFIG_ERR_MASK **0x19CCC**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	ERR	RT	External memory is allocated but not available

Table 87 Device Configuration Error Register

Notes

#	Signal	Stat	Description
7-1	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 87 Device Configuration Error Register

Note:

0) Error - This flag indicates that the SerB is programmed to have more memory than is actually attached to P-Port.

8.5.5 sRIO DMA Status Register

These flags are generated whenever an sRIO error occurs that is not covered by one of the error flags defined in the sRIO specification. These errors are monitored by the Case Scenarios, so there is one flag register per case scenario.

Name: **CS0_DMA_STAT_FLAG** **Address:** **0x19C10**
CS0_DMA_STAT_MASK **0x19CD0**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	MEMSTOP	RT	sRIO NEXT ADDRESS has reached STOP ADDRESS
1	TALLY1	RT	The packet tally counter wrapped on S-Port
7-2	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 88 sRIO DMA Status Register

Notes

Note:

0) Memory Stop - The sRIO memory address has incremented to or beyond the stop address. This flag may be used to send a doorbell if the address reaches the stop address (triggering the flag condition), the flag will remain active until software writes a "1" to clear the flag.

1) Tally1 - The packet tally counter will wrap. If the user whishes to know it wrapped, the flag may be used.

8.5.6 Missing 2 Packet Flag Register

If missing 2 packet is turned on and two or more packets are missing, the flags of this register will be used. Note that If this register is read and cleared, the "Missing Packet Address Logging Register 1" will also be cleared.

Name: **MISS2_PKT_FLAG** **Address:** **0x19C50**
MISS2_PKT_MASK **0x19D10**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	MISSIN2	CL	Two or more sRIO packets were detected as missing
7-1	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 89 Missing Packet Flag Register

Note:

0) Missing 2 - If two or more packets are missing, they cannot be replaced. This flag indicates that a catastrophic error has occurred in the PPS application.

8.5.7 FIFO Empty Flag Register

If this register generates an sRIO packet, the packet will be a doorbell.

Name: **FIFO_EMPTY_FLAG** **Address:** **0x19C60**
FIFO_EMPTY_MASK **0x19D20**

#	Signal	Stat	Description
0	EF	RT	Queue 0, Empty Flag
1	PAE	RT	Queue 0, Programmable Almost Empty
2	PR	RT	Queue 0, Packet Ready
3	W	RT	Queue 0, Waterlevel Exceeds Packet Count
7-4	-		Reserved

Table 90 FIFO Queue Empty Flag Register

Notes

#	Signal	Stat	Description
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 90 FIFO Queue Empty Flag Register

Note:

0) EF = No data remains in queue
1) PAE = Programmable on 1/256th portions of queue memory
2) PR = Full packet ready for reading, reading one byte or more will kill flag
3) W = Waterlevel is at or past the watermark (either byte or packet count)
7-4) Res = Reserved bit

8.5.8 FIFO Full Flag Register

If this register generates an sRIO packet, the packet will be a doorbell.

Name: **FIFO_FULL_FLAG** **Address:** **0x19C64**
FIFO_FULL_MASK **0x19D24**

#	Signal	Stat	Description
0	FF	RT	Queue 0, Full Flag, Incoming packet rejected
1	PAF	RT	Queue 0, Programmable Almost Full
2	SA	RT	Queue 0, One or more Max Sized Packet Space Available
7-3	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DestID	RW	Destination ID for sRIO Doorbell
31-24	DestID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port 1 Doorbell Mask
47-40	-		Unused bits
55-48	MASK	RW	Interrupt 0 Mask
63-56	MASK	RW	Interrupt 1 Mask

Table 91 FIFO Queue Full Flag Register

Notes

Note:

0) FF = No space remains in queue, the entire incoming packet was rejected

1) PAF = Programmable on 1/256th portions of queue memory

2) SA = Space available for one full sized packet. A doorbell will be sent whenever the flag changes state. Flag is inactive whenever the incoming packet may prevent an additional packet entering.

8.5.9 DSP Interrupt Flag Register

In the TI application, if a doorbell is sent to the DSP, it must have a programmable content. To solve the problem, unmasked flags may send a doorbell to the DSP upon toggling. When enabled, every unmasked flag (except the packet tally flags) will send a doorbell with the programmed content to the DSP at the programmed destination ID.

This register sends an sRIO doorbell on S-Port and is not capable of generating an interrupt on Int 0 or Int 1. Since it has programmable content, there is no register number associated with this register.

Name: **DSP_INT_FLAG** **Address:** **0x19CA0**
DSP_INT_MASK **0x19E60**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	FLAG	RT	The Aggregate of all unmasked flags
7-1	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
63-48	POINTER	RW	Programmable Doorbell Contents

Table 92 DSP Interrupt Flag Register

Note:

0) This flag will toggle if any other unmasked flag toggles. This flag may be cleared only by clearing the source flag.

24:16) This is the destination ID for the DSP that must receive the programmed doorbell. This may be any destination.

39:32) The doorbell is turned on and off by masking bit 0. This mask does not affect the source flags. The source flags each have their own flags and masks. The Int 1 mask associated with each individual flag is used to enable the flags to toggle the DSP Interrupt Flag. The tally flag should always be masked off in Int 1 masks on the individual flag registers to avoid toggling the DSP Interrupt Flag. Int 1 and the DSP mask share a mask and must toggle due to the same flags.

63:48) The content of the doorbell is a 16 bit user programmable pointer.

8.5.10 Tally Doorbell Flag Register

If the "Tally 1" flag in the sRIO DMA Status Information Register toggles in the TI application, the user may wish to send a doorbell to the DSP. The content of the doorbell is user programmable, to allow the user to select an interrupt within the DSP. This doorbell does not interfere with any doorbells or interrupts generated by the sRIO DMA Status Information Register.

Notes

This register sends an sRIO doorbell on S-Port and is not capable of generating an interrupt on Int 0 or Int 1, therefore there is no register number.

Name: **CS0_TALY_DBEL_FLAG** **Address:** **0x19E10**
CS0_TALY_DBEL_MASK **0x19ED0**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	-		Reserved
1	TALLY1	CL	The Packet Tally Counter Wrapped on S-Port
7-2	-		Unused bits
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	S-Port Doorbell Mask
47-40	-		Unused bits
63-48	POINTER	RW	Programmable Doorbell Contents

Table 93 Tally Doorbell Flag Register

Note:

1) This flag is identical to the flag in the "sRIO DMA Status Information" register.

24:16) This is the destination ID for the DSP that must receive the tally indication doorbell. It is not required that this be the same destination as in the "sRIO DMA Status Information" register.

39:32) The doorbell is turned on and off by masking bit 1.

63:48) The content of the doorbell is a 16 bit user programmable pointer.

8.5.11 Missing 2 Packet Programmable Flag Register

If missing 2 packet is turned on and two or more packets are missing, this register will allow a programmable doorbell to be sent to a designated sRIO destination ID. Note that If this register is read and cleared, the “Missing Packet Address Logging Register 2” will also be cleared.

Name: **MISS2_PGRM_FLAG** **Address:** **0x19E50**
MISS2_PGRM_MASK **0x19F10**

If this register generates an sRIO packet, the packet will be a doorbell.

#	Signal	Stat	Description
0	MISSIN2	CL	Two or more sRIO packets were detected as missing
7-1	-		Reserved
8	TT	RW	Defines whether the sRIO doorbell is an 8 or 16 bit destination ID
9	WR32	RW	0 = Write 8 bits (clear flags only), 1 = Write 32 bits (write new dest IDs)
11-10	PRIO	RW	Priority for Doorbell packet

Table 94 Missing Packet Programmable Flag Register

Notes

#	Signal	Stat	Description
15-12	-		Unused bits
23-16	DESTID	RW	Destination ID for sRIO Doorbell
31-24	DESTID	RW	Destination ID for sRIO Doorbell, for 16 bit extension
39-32	MASK	RW	Programmable sRIO Doorbell Mask
47-40	-		Unused bits
63-48	POINTER	RW	Programmable Doorbell Contents

Table 94 Missing Packet Programmable Flag Register

Note:

0) Missing 2 - If two or more packets are missing, they cannot be replaced. This flag indicates that a catastrophic error has occurred in the PPS application. The programmable contents of locations 63-48 will be sent to the destination ID in locations 31-16.

Notes

9.0 Reset and Initialization

The SerB does not require specific power sequencing between any of the core and I/O supplies.

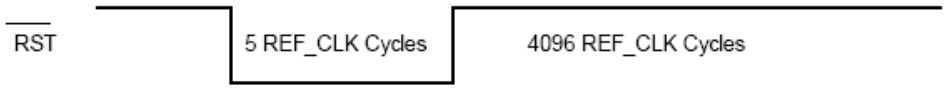


Figure 11 Reset Timeline

To reset the device, first reset signal has to be de-asserted (Reset Low), and it is asserted after 5 REF_CLK cycles. 4096 REF_CLK cycles later, the device completes the reset process. Once completed, access to the SerB from any and all interfaces is possible and the SerB is fully functional. COntrol and data traffic will not be accepted by the SerB until this process is fully completed.

9.1 Speed Select (SPD[1:0])

There are 2 port speed select pins. These pins are used to chose the initial speed on sRIO ports. The selection table is given below:

Value on the Pins (SPD1 SPD0)	Ports Rate
00	1.25Gbps
01	2.5Gbps
10	3.125Gbps
11	Reserved

Table 95 Port Speed Selection Pin Values

9.2 sRIO Reset Control Symbol

The sRIO Reset Control Symbol is defined by the RIO spec to perform a master reset on the target device. It is a link level reset and must be received four times to perform the reset. Despite it being a control symbol generated at the link level, the use of the reset is generally instructed from higher-level authority than the link.

The PPS has taken the control symbol and has allowed the user to program the severity of the reset either as a full master reset, or as an sRIO port reset only. The PPS also has the capability of receiving instructions from the DSP to send a control symbol on any one of its ports to reset other attached devices.

The SerB will not have the capabilities of the PPS and will perform only a full Master Reset whenever the sRIO reset control symbol has been received four times. The count of four will reset whenever a packet other than an sRIO reset control symbol is successfully received. The control symbol has no capability to form anything other than a Master Reset. Any other sRIO resets must be received in the form of type 8 maintenance packets.

More details on the control symbol can be found in section 3.4.5.1 of Physical Layer x1/x4 LP-Serial Specification.

9.3 JTAG Reset

At Power-Up, $\overline{\text{TRST}}$ must be asserted LOW to bring the TAP controller up in a known, reset state. Per IEEE 1149.1 specification, the user can alternatively hold TMS pin high while clocking TCK five times (minimum) to reset the controller.

To deactivate JTAG, $\overline{\text{TRST}}$ should be tied low so that the TAP controller remains in a known state at all times. All of the other JTAG input pins are internally biased in such a way that by leaving them unconnected they are automatically disabled. Note that JTAG inputs are OK to float because they have leakers (as required by IEEE 1149.1 specification).

9.4 System Initialization

The SerB will automatically configure itself upon power up to the default configuration set by the hard-wired inputs. For the duration of the default configuration, the SerB will not accept packets on either serial port. Once the SerB has achieved the default configuration, the ports will become active and may accept data. If additional programming is to be completed

Notes

after the default configuration is active, the user should be aware that data or additional configuration information might be accepted on multiple ports. The user must exercise care to insure that the incoming configuration and data does not interfere with the device programming. In many cases, a partial reset may clear unwanted data, but may cause corruption of active transfers.

Before operation, the SerB must be configured. The steps of configuration are as follows:

- Power on. No power sequencing is required, but all power supplies must have achieved the minimum required level before proceeding.
- Master reset may be applied at any time. If reset is performed in association with Power on, reset may be applied before, during or after Power On, but the reset must be held after achieving valid power levels for the designated minimum number of clock cycles (defined in the electrical section).
- SerB will initialize itself according to the hard-wired pins.
  - The PLLs will take time to lock
  - The PHYs will begin to negotiate with neighboring devices, attempting to establish links
  - The memory will be allocated, per the default configuration
- I^2C may be used for additional programming if required without waiting for PLL lock. All of the configuration registers may be programmed through I^2C.
- JTAG has access to the configuration registers. If JTAG is not used for additional programming, the JTAG inputs should be disabled. Full operation of JTAG is described in the JTAG section.
- In the event that the SerB needs to be programmed over a serial port, the serial port must have achieved full Link Up status before programming may commence.
- If interrupt masks are needed, the masks should be programmed using one of the programming methods.
- After programming, the SerB should be fully functional. It should be noted that the SerB may be reconfigured at any time. It should be noted that at any time, the SerB may reconfigured through I^2C or a Serial Port.

9.5 Initialization of RIO Ports

The sRIO ports shall be initialized before they are operational. More needs to be developed on this topic, but as a start, the following references should be made:

- PPS specification an sRIO port Initialization
- RIO Physical Layer Specification section 4.6
- RIO System Bring-up document for explanation and examples of system bring-up
- Hard-wired pin description

The initialization of the sRIO ports may be influenced by the following sources:

- sRIO maintenance packets
- I^2C programming
- JTAG programming
- Hard-wired inputs

Notes

10.0 Reference Clock

There are several clocks associated with the SerB. All internal operational clocks within the SerB are generated from the PHY Reference clock operating at 156.25M Hz. The following clocks are used within the SerB:

1. PHY Reference Clock. This clock is an input at 156.25M Hz and is used to drive the serial ports and internal functions. When P-Port is a QDR memory port, the PHY Reference clock also drives the memory interface.
2. JTAG Clock
3. I^2C Clock

10.1 Reference Clock Electrical Specifications

The reference clock is 156.25 MHz, and is AC-coupled with the following electrical specifications:

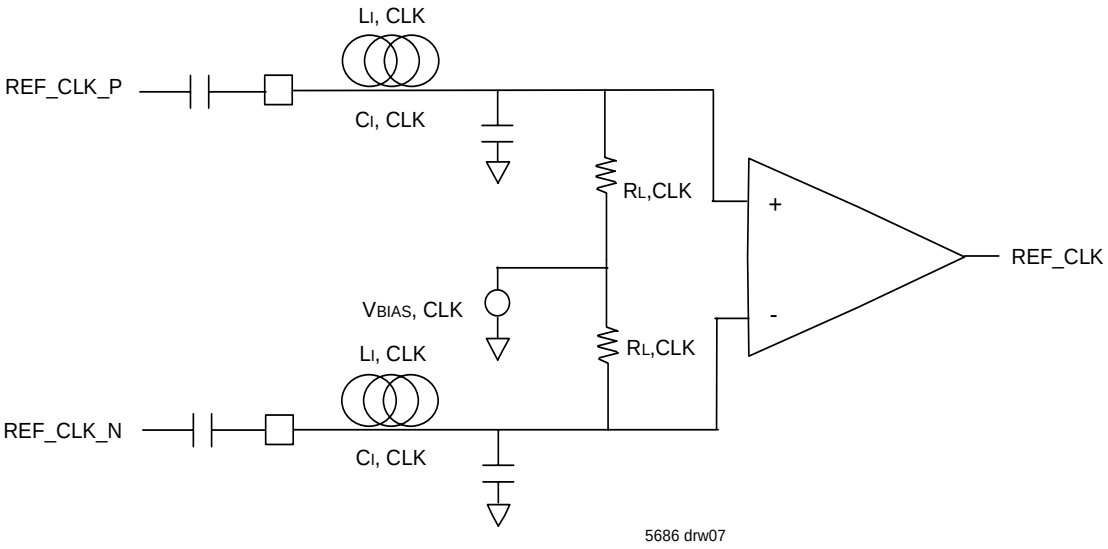


Figure 12 REF_CLK representative circuit

Name	Description	Min	Nom	Max	Units
REF_CLK	REF_CLK clock running at 156.25Mhz	-100	----	+100	ppm
tDUTY_REF	REF_CLK duty cycle	40	50	60	%
tRCLK/tFCLK	Input signal rise/fall time (20%-80%)	200	500	650	ps
vIN_CML	Differential peak-peak REF_CLK input swing	400	----	2400	mV
RL_CLK	Input termination resistance	40	50	60	ohm
LI_CLK	Input inductance	-----	-----	4	nH
CI_CLK	Input capacitance	-----	-----	5	pF

Table 96 Input Reference Clock Jitter Specifications

The reference clock wander should not be more than 100ppm (for 156.25 Mhz, this is +/-15.625 KHz). This requirement comes from the sRIO specification that outgoing signals from separate links, which belong to the same port, should not be separated more than 100ppm.

Note that the series capacitors are descretes that must be placed external to the devices's receivers. All other elements are associated with the input structure internal to the device. V_{BIAS} is generated internally.

Notes

11.0 Absolute Maximum Ratings(1)

Symbol	Rating	Commercial & Industrial	Unit
V_{TERM} ($V_{DD}3$)	$V_{DD}3$ Terminal Voltage with Respect to GND	-0.5 to 3.6	V
V_{TERM}(2) ($V_{DD}3$-supplied interfaces)	Input or I/O Terminal Voltage with Respect to GND	-0.3 to $V_{DD}3$+0.3	V
V_{TERM} (V_{DD})	V_{DD} Terminal Voltage with Respect to GND	-0.5 to 1.5	V
V_{TERM}(2) (V_{DD}-supplied interfaces)	Input or I/O Terminal Voltage with Respect to GND	-0.3 to V_{DD}+0.3	V
V_{TERM} (V_{DDS})	V_{DDS} Terminal Voltage with Respect to GNDS	-0.5 to 1.5	V
V_{TERM}(2) (V_{DDS}-supplied interfaces)	Input or I/O Terminal Voltage with Respect to GNDS	-0.3 to V_{DDS}+0.3	V
V_{TERM} (V_{DDA})	V_{DDA} Terminal Voltage with Respect to GNDS	-0.5 to 1.5	V
V_{TERM}(2) (V_{DDA}-supplied interfaces)	Input or I/O Terminal Voltage with Respect to GNDS	-0.3 to V_{DDA}+0.3	V
T_{BIAS}(3)	Temperature Under Bias	-55 to +125	C
T_{STG}	Storage Temperature	-65 to +150	C
T_{JN}	Junction Temperature	+150	C
I_{OUT} (For $V_{DD}3$ = 3.3V)	DC Output Current	30	mA
I_{OUT} (For $V_{DD}3$ = 2.5V)	DC Output Current	30	mA

Table 97 Absolute Maximum Ratings

Note:

1. Stresses greater than those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.
2. This is a steady-state DC parameter that applies after the power supply has reached its nominal operating value. Power sequencing is not necessary; however, the voltage on any Input or I/O pin cannot exceed its corresponding supply voltage during power supply ramp up.
3. Ambient Temperature under DC Bias. No AC Conditions.

Notes

11.1 Recommended Temperature and Operating Voltage[1]

Grade	Ambient Temperature	Max Junction Temperature (T_{JN})	Ground[2]	Supply Voltage[4]
Commercial	0°C to 70'C	125°C	GND = 0V GNDS = 0V	V_{DD} = 1.2 +/- 5% V_{DDQ} = 1.5 +/- 5% V_{DDS} = 1.2 +/- 5% V_{DD3}[3] = 3.3 +/- 5%, or 2.5V +/- 100mV V_{DDA} = 1.2 +/- 5%
Industrial	-40°C to 85°C	125'C	GND = 0V GNDS = 0V	V_{DD} = 1.2 +/- 5% V_{DDQ} = 1.5 +/- 5% V_{DDS} = 1.2 +/- 5% V_{DD3}[3] = 3.3 +/- 5%, or 2.5V +/- 100mV V_{DDA} = 1.2 +/- 5%

Table 98 Recommended Temperature and Operating Voltage

Note:

1. Power sequencing is not necessary; however, the voltage on any Input or I/O pin cannot exceed its corresponding supply voltage during power supply ramp up. The device is not sensitive to supply rise and fall times, and thus these are not specified.
2. V_{DD3}, V_{DDA}, and V_{DDS} share a common ground (GNDS). Core supply and ground are V_{DD} and GND respectively.
3. V_{DD3} may be operated at either 3.3V or 2.5V simply by providing that supply voltage. For those interfaces operating on this supply, this datasheet provides input and output specifications at each of these voltages.
4. V_{DDS} & V_{DDA} may be tied to a common plane. V_{DD} (core, digital supply) should have its own supply and plane.

11.2 AC Test Conditions

Input Pulse Levels	GND to 3.0V / GND to 2.4V
Input Rise / Fall Times	2ns
Input Timing Reference Levels	1.5V / 2.5V
Output Reference Levels	1.5V / 1.25V
Output Load	Figure 12

Table 99 AC Test Conditions (V_{DD3} = 3.3V / 2.5V): JTAG, I^2C, RST

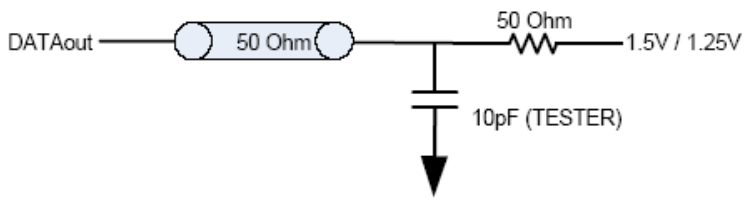


Figure 13 AC Output Test Load (JTAG)

Notes

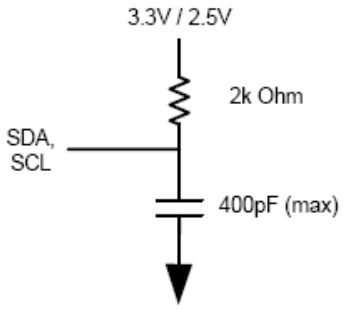


Figure 14 AC Output Test Load (I^2C)

Note:

1. The SDA and SCL pins are open-drain drivers. Refer to the Philips I^2C Specification [1] for appropriate selection of pull-up resistors for each.

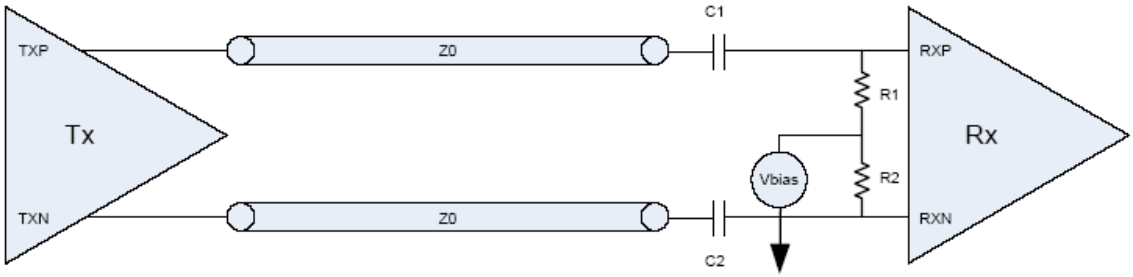


Figure 15 sRIO Lanes Test Load

Note:

1. The characteristic impedance Z0 should be designed for 100 ohms. An in line capacitor C1 and C2 at each input of the receiver provides AC-coupling and a DC-block. The IST recommended and test value is 100nF for each. Thus, ant DC bias differential between the two devices on the link is negated. The differential input resistance is designed to be 100 Ohms (per sRIO specification). Thus, R1 and R2 are 50 Ohms each. Note that V_{BIAS} is the internal bias voltage of the device's receiver.

11.3 Typical Power Figures

Typical Power Draw	IDD	IDDS	IDDA	IDDQ	IDD3
4.5W (all supplies)	2.0A	460mA	216mA	465mA	30mA

Table 100 Typical Power Figures

Note:

1. Values are based on characterization, and are not production tested.

Notes

12.0 I²C-Bus

The SerB is compliant with the I²C specification [1]. This specification provides all functional detail and electrical specifications associated with the I²C bus. This includes signaling, addressing, arbitration, AC timing, DC specifications, and other details.

The I²C bus is comprised of Serial Data (SDA) and Serial Clock (SCL) pins and can be used to attach a CPU for initialization and management purposes. A CPU can then access registers and program the device, but it cannot access other devices attached to the sRIO interfaces through the I²C bus. The I²C interface supports Fast/Standard (F/S) mode (400/100 kHz). The SerB does NOT support CBUS or General Address calls.

12.1 I²C Device Address

Relative to I²C, the SerB is a slave-only receiver and transmitter. The device address for the SerB is fully pin-defined by 10 external pins. This provides full flexibility in defining the slave address to avoid conflicting with other I²C devices on a given bus. The SerB may be operated as either a 10-bit addressable device or a 7-bit addressable device based on another external pin Address Select (ADS). If the ADS pin is tied to Vdd, then the SerB operates as a 10-bit addressable device and the device address will be defined as ID[9:0]. If the ADS pin is tied to GND, then the SerB operates as a 7-bit addressable device with the device address defined by ID[6:0]. The addressing mode must be established at power-up and remain static throughout operation. Dynamic changes will result in undetermined behavior.

Pin	I²C Address Bit (pin_addr)
ID0	0
ID1	1
ID2	2
ID3	3
ID4	4
ID5	5
ID6	6
ID7	7 (don't care in 7-bit mode)
ID8	8 (don't care in 7-bit mode)
ID9	9 (don't care in 7-bit mode)

Table 101 I²C static address selection pin configuration

All of the SerB's registers are addressable through I²C. These registers are accessed via 22-bit addresses and 32-bit word boundaries though standard reads and writes. These registers may also be accessed through the sRIO and JTAG interfaces.

12.2 Signaling

The SerB is a slave-only receive and transmit device. Thus, communication with the SerB on the I²C bus follows these two cases:

1. Suppose a master device wants to send information to the SerB:
   - Master device addresses SerB (slave)
   - Master device (master-transmitter), sends data to SerB (slave- receiver)
   - Master device terminates the transfer
2. If a master device wants to receive information from the SerB:
   - Master device addresses SerB (slave)
   - Master device (master-receiver) receives data from SerB (slave- transmitter)
   - Master device terminates the transfer.

All signaling is fully compliant with I²C. Full detail of signaling can be found in the I²C specification [1].

Notes

12.2.1 Interfacing to Standard-, Fast-, and Hs-mode devices

The SerB supports Fast / Standard (F/S) modes of operation. Per I^2C specification, in mixed speed communication the SerB supports Hs- and Fast-mode devices at 400 kbit/s, and Standard-mode devices at 100 kbit/s. Please refer to the I^2C specification for detail on speed negotiation on a mixed speed bus.

12.2.2 SerB Specific Memory Access

There is a SerB-specific I^2C memory access implementation. This implementation is fully I^2C compliant. It requires the memory address to be explicitly specified during writes. This provides directed memory accesses through the I^2C bus. Subsequent reads always begin at the address specified during the last write.

The write procedure requires the 3-Bytes (22-bits) of memory address to be provided following the device address. Thus, the following are required: device address – one or two bytes depending on 10-bit/7-bit addressing, memory address – 3 bytes yielding 22-bits of memory address, and a 32-bit data payload – 4 byte words.

The read procedure has the memory address section of the transfer removed. Thus, to perform a read, the proper access would be to perform a write operation and issue a repeated start after the acknowledge bit following the third byte of memory address. Then, the master would issue a read command selecting the SerB through the standard device address procedure with the R/W bit high. Note that in 10-bit device address mode (ADS=1), only the two MSBs need be provided during this read. Data from the previously loaded address would immediately follow the device address protocol. It is possible to issue a stop or repeated start anytime during the write data payload procedure, but must be before the final acknowledge (i.e. canceling the write before the actual write operation is completed and performed). Also, the master would be allowed to access other devices attached to the I^2C bus before returning to select the SerB for the subsequent read operation from the loaded address.

12.3 Figures

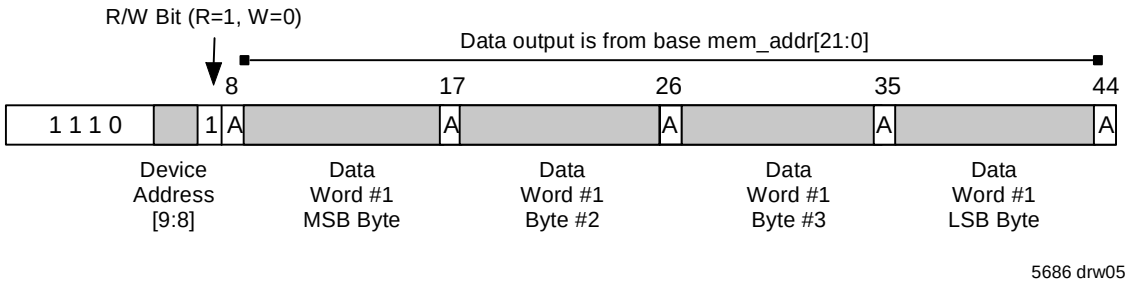


Figure 16 Write protocol with 10-bit Slave Address (ADS =1)

Note:

1. I^2C writes to memory align on 32-bit word boundaries, thus the 22 address MSBs must be provided while the 2 LSBs associated with word and byte pointers are DON'T CARE and are therefore not transmitted.

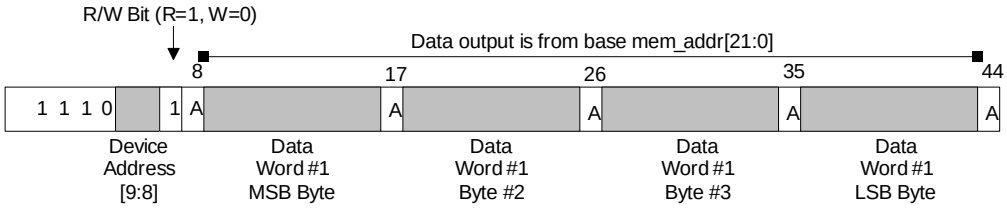


Figure 17 Read Protocol with 10-bit Slave Address (ADS=1)

Notes

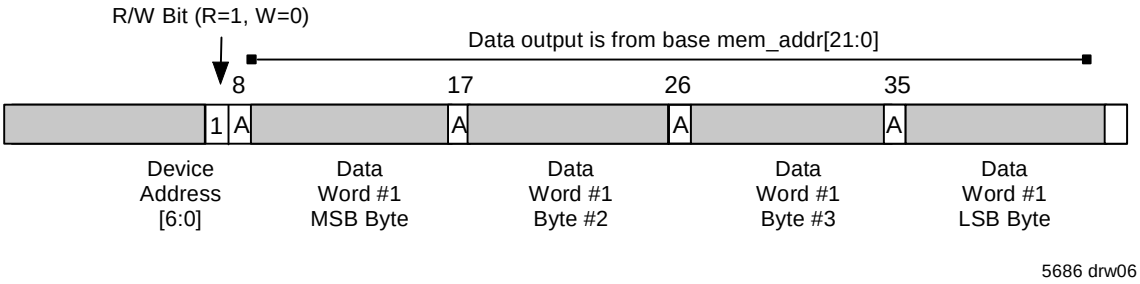


Figure 18 Write protocol with 7-bit Slave Address (ADS=0)

Note:

1. I^2C writes to memory align on 32-bit word boundaries, thus the 22 address MSBs must be provided while the 2 LSBs associated with word and byte pointers are DON'T CARE and are therefore not transmitted.

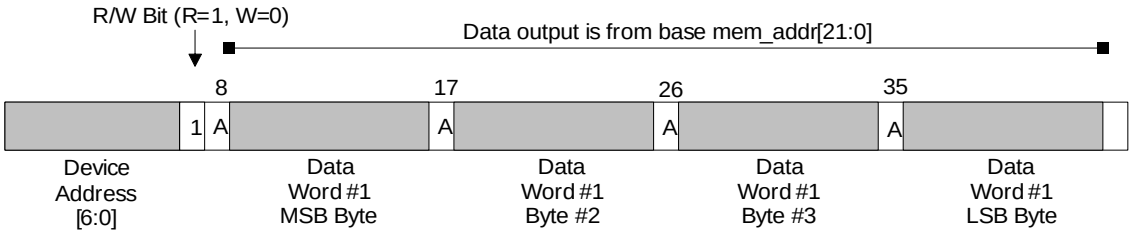


Figure 19 Read protocol with 7-bit Slave Address (ADS=0)

12.4 I^2C DC Electrical Specifications

Note that the ADS and ID pins will all run off the core (1.2V) power supply, and these pins are required to be fixed during operation. Thus, these pins must be statically tied to the 1.2V supply or GND.

Tables 19 and 20 below lists the SDA and SCL electrical specifications for F/S-mode I^2C devices:

At recommended operating conditions with V_{DD3} = 3.3V ± 5%

Parameter	Symbol	Min	Max	Unit
Input high voltage level	V_{IH}	0.7 x V_{DD3}	$V_{DD3\,(MAX)}$+ 0.5	V
Input low voltage level	V_{IL}	-0.5	0.3 x V_{DD3}	V
Hysteresis of Schmitt trigger inputs:	Vhys	0.05 x V_{DD3}	-	
Low level output voltage	V_{OL}	0	0.2 x V_{DD3}	V
Output fall time from $V_{IH(MIN)}$ to $V_{IL(MAX)}$ with a bus capacitance from 10pF to 400pF	t_{OF}	20 + 0.1 x C_b	250	ns
Pulse width of spikes which must be suppressed by the input filter	t_{SP}	0	50	ns
Input current each I/O pin (input voltage is between 0.1 x V_{DD3} and 0.9 x $V_{DD3\,(MAX)}$)	I_I	-10	10	uA
Capacitance for each I/O pin	C_I	-	10	pF

Figure 20 I^2C SDA & SCL DC Electrical Specifications

Notes

At recommended operating conditions with V_{DD3} = 2.5V ± 100mV

Parameter	Symbol	Min	Max	Unit
Input high voltage level	V_{IH}	0.7 x V_{DD3}	$V_{DD3(MAX)}$ + 0.1	V
Input low voltage level	V_{IL}	-0.5	0.3 x V_{DD3}	V
Hysteresis of Schmitt trigger inputs:	Vhys	0.05 x V_{DD3}	-	
Low level output voltage	V_{OL}	0	0.2 x V_{DD3}	V
Output fall time from $V_{IH(MIN)}$ to $V_{IL(MAX)}$ with a bus capacitance from 10pF to 400pF	t_{OF}	20 + 0.1 x C_b	250	ns
Pulse width of spikes which must be suppressed by the input filter	t_{SP}	0	50	ns
Input current each I/O pin (input voltage is between 0.1 x V_{DD3} and 0.9 x $V_{DD3\ (MAX)}$)	I_I	-10	10	uA
Capacitance for each I/O pin	C_I	-	10	pF

Figure 21 I²C SDA & SCL DC Electrical Specifications

12.5 I²C AC Electrical Specifications

Signal	Symbol	Reference Edge	Standard Mode		Fast Mode		Unit
			Min	Max	Min	Max	
$I^2C^{(1,4)}$							
SCL	f_{SCL}	none	0	100	0	400	kHz
	$t_{HD;STA}$		4.0	—	0.6	—	µs
	t_R		—	1000	—	300	µs
	t_F		—	300	—	300	µs
$SDA^{(2,3)}$	$t_{SU;DAT}$	SCL rising	250	—	100	—	µs
	$t_{HD;DAT}$		0	3.45	0	0.9	µs
	t_R		—	1000	10	300	µs
	t_F		—	300	10	300	µs
Start or repeated start condition	$t_{SU;STA}$	SDA falling	4.7	—	0.6	—	µs
	$t_{SU;STO}$		4.0	—	0.6	—	µs
Stop condition	$t_{SU;STO}$	SDA rising	4.0	—	0.6	—	µs
Bus free time between a stop and start condition	t_{BUF}		4.7	—	1.3	—	µs
Capacitive load for each bus line	C_b		—	400	—	400	pF

Figure 22 Specifications of the SDA and SCL bus lines for F/S-mode I²C -bus devices

Note:

1. For more information, see the I2C-Bus specification by Philips Semiconductor [1].
2. A device must internally provide a hold time of at least 300 ns for the SDA signal (referred to the V_{IHMIN} of the SCL signal) to bridge the undefined region of the falling edge of SCL.
3. The maximum $t_{HD;DAT}$ has only to be met if the device does not stretch the LOW period (t_{LOW}) of the SCL signal.
4. A Fast-mode I²C-bus device can be used in a Standard-mode I²C-bus system, but the requirement $t_{SU;DAT} \geq$ 250 ns must then be met. This will automatically be the case if the device does not stretch the LOW period of the SCL signal. If such a device does stretch the LOW period of the SCL signal, it must output the next data bit to the SDA line $t_{RMAX} + t_{SU;DAT}$ = 1000 + 250 = 1250 ns (according to the Standard-mode I²C-bus specification) before the SCL line is released.

Notes

12.6 I2C Timing Waveforms

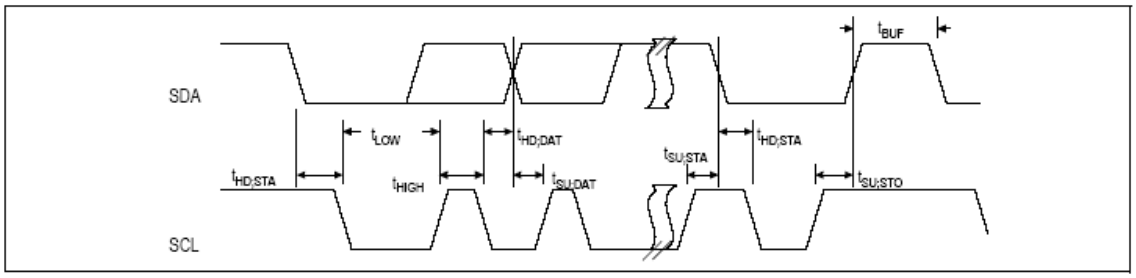


Figure 23 I2C Timing Waveforms

Notes

Notes

13.0 Serial RapidIO™ AC Specifications

13.1 Overview

The SerB's SERDES are in full compliance to the RapidIO™ AC specifications for the LP-Serial physical layer [5]. This section provides those specifications for reference. The electrical specifications cover both single and multiple-lane links. Two transmitters (short run and long run) and a single receiver are specified for each of three baud rates, 1.25, 2.50, and 3.125 GBaud.

Two transmitter specifications allow for solutions ranging from chip-to-chip interconnect to driving two connectors across a backplane. A single receiver specification is given that will accept signals from both the short run and long run transmitter specifications.

The short run transmitter should be used mainly for chip-to-chip connections on either the same printed circuit board or across a single connector. This covers the case where connections are made to a mezzanine (daughter) card. The minimum swings of the short run specification reduce the overall power used by the transceivers.

The long run transmitter specifications use larger voltage swings that are capable of driving signals across backplanes. This allows a user to drive signals across two connectors and a backplane. The SerB can drive beyond the specification distance of at least 50 cm at all baud rates. Please use IDT's Simulation Kit IO models to determine reach and signal quality for a given PCB design.

All unit intervals are specified with a tolerance of +/- 100 ppm. The worst case frequency difference between any transmit and receive clock will be 200 ppm.

To ensure inter-operability between drivers and receivers of different vendors and technologies, AC coupling at the receiver input must be used.

13.2 Signal Definitions

LP-Serial links uses differential signaling. This section defines terms used in the description and specification of differential signals. Differential Peak-Peak Voltage of Transmitter or Receiver shows how the signals are defined. The figure below shows waveforms for either a transmitter output (TD and $\overline{TD}$) or a receiver input (RD and $\overline{RD}$). Each signal swings between A Volts and B Volts where A > B. Using these waveforms, the definitions are as follows:

1. The transmitter output signals and the receiver input signals TD,$\overline{TD}$, RD and $\overline{RD}$ each have a peak-to-peak swing of A - B Volts
2. The differential output signal of the transmitter, V_{OD}, is defined as V_{TD}-$V_{\overline{TD}}$.
3. The differential input signal of the receiver, V_{ID}, is defined as V_{RD}-$V_{\overline{RD}}$.
4. The differential output signal of the transmitter and the differential input signal of the receiver each range from A - B to -(A - B) Volts.
5. The peak value of the differential transmitter output signal and the differential receiver input signal is A - B Volts
6. The peak-to-peak value of the differential transmitter output signal and the
7. Differential receiver input signal is 2 * (A - B) Volts

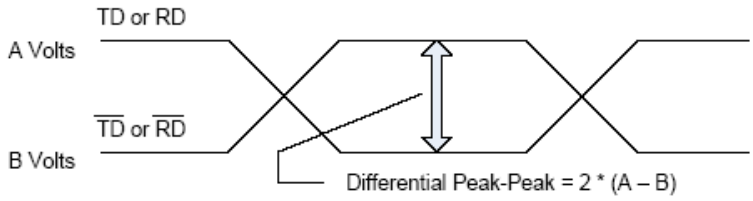


Figure 24 Differential Peak-Peak Voltage of Transmitter or Receiver

Notes

To illustrate these definitions using real values, consider the case of a CML (Current Mode Logic) transmitter that has a common mode voltage of 2.25 V and each of its outputs, TD and $\overline{TD}$, has a swing that goes between 2.5V and 2.0V. Using these values, the peak-to-peak voltage swing of the signals TD and $\overline{TD}$ is 500 mV p-p. The differential output signal ranges between 500 mV and -500 mV. The peak differential voltage is 500 mV. The peak-to-peak differential voltage is 1000 mV p-p.

13.3 Equalization

With the use of high speed serial links, the interconnect media will cause degradation of the signal at the receiver. Effects such as Inter-Symbol Interference (ISI) or data dependent jitter are produced. This loss can be large enough to degrade the eye opening at the receiver beyond what is allowed in the specification. To negate a portion of these effects, equalization can be used. The equalization technique implemented in the SerB is Pre-emphasis on the transmitter (under register control)

13.4 Explanatory Note on Transmitter and Receiver Specifications

AC electrical specifications are given for transmitter and receiver. Long run and short run interfaces at three baud rates (a total of six cases) are described.

The parameters for the AC electrical specifications are guided by the XAUI electrical interface specified in Clause 47 of IEEE 802.3ae-2002.

XAUI has similar application goals to serial RapidIO™. The goal of this standard is that electrical designs for serial RapidIO™ can reuse electrical designs for XAUI, suitably modified for applications at the baud intervals and reaches described herein.

13.5 Transmitter Specifications

LP-Serial transmitter electrical and timing specifications are stated in the text and tables of this section.

The differential return loss, S11, of the transmitter in each case shall be better than

-10 dB for (Baud Frequency)/10 < Freq(f) < 625 MHz, and

-10 dB + 10log(f/625 MHz) dB for 625 MHz <= Freq(f) <= Baud Frequency

The reference impedance for the differential return loss measurements is 100 Ohm resistive. Differential return loss includes contributions from on-chip circuitry, chip packaging and any off-chip components related to the driver. The output impedance requirement applies to all valid output levels.

The 80KSBR200 satisfies the specification requirement that the 20%-80% rise/fall time of the transmitter, as measured at the transmitter output, in each case have a minimum value 60 ps.

Similarly the timing skew at the output of an LP-Serial transmitter between the two signals that comprise a differential pair not exceed 25 ps at 1.25 GB, 20 ps at 2.50 GB and 15 ps at 3.125 GB.

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal com-prising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	500	1000	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	800	800	ps	+/- 100 ppm

Figure 25 Short Run Transmitter AC Timing Specifications - 1.25 GBaud

Notes

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal comprising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	500	1000	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	400	400	ps	+/- 100 ppm

Figure 26 Short Run Transmitter AC Timing Specifications - 2.5 GBaud

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal comprising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	500	1000	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	320	320	ps	+/- 100 ppm

Figure 27 Short Run Transmitter AC Timing Specifications - 3.125 GBaud

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal comprising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	800	1600	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	800	800	ps	+/- 100 ppm

Figure 28 Long Run Transmitter AC Timing Specifications - 1.25 GBaud

Notes

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal comprising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	800	1600	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	400	400	ps	+/- 100 ppm

Figure 29 Long Run Transmitter AC Timing Specifications - 2.5 GBaud

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COM-MON of either signal comprising a differential pair.
$V_{DIFF\ PP}$	Differential Output Voltage	800	1600	mV p-p	
J_D	Deterministic Jitter	-	0.17	UI p-p	
J_T	Total Jitter	-	0.35	UI p-p	
S_{MO}	Multiple Output Skew	-	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	320	320	ps	+/- 100 ppm

Figure 30 Long Run Transmitter AC Timing Specifications - 3.125 GBaud

For each baud rate at which an LP-Serial transmitter is specified to operate, the output eye pattern of the transmitter shall fall entirely within the un-shaded portion of the Transmitter Output Compliance Mask shown in Figure 30 with the parameters specified in Figure 31. The eye pattern is measured at the output pins of the device and the device is driving a 100 Ohm +/- 5% differential resistive load. The output eye pattern of a LP-Serial transmitter that implements pre-emphasis (to equalize the link and reduce inter-symbol interference) need only comply with the Transmitter Output Compliance Mask when pre-emphasis is disabled or minimized.

Notes

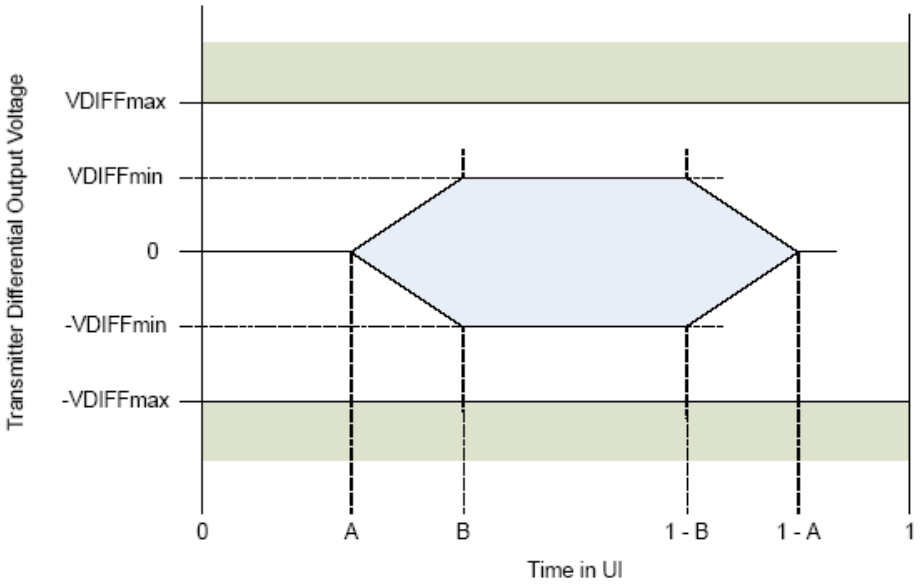


Figure 31 Transmitter Output Compliance Mask

Transmitter Setting	$V_{DIFFmin}$ (mV)	$V_{DIFFmax}$ (mV)	A (UI)	B (UI)
1.25 GBaud Short Range	250	500	0.175	0.39
1.25 GBaud Long Range	400	800	0.175	0.39
2.5 GBaud Short Range	250	500	0.175	0.39
2.5 GBaud Long Range	400	800	0.175	0.39
3.125 GBaud Short Range	250	500	0.175	0.39
3.125 Gbaud Long Range	400	800	0.175	0.39

Figure 32 Transmitter Differential Output Eye Diagram Parameters

13.6 Receiver Specifications

LP-Serial receiver electrical and timing specifications are stated in the text and tables of this section.

Receiver input impedance shall result in a differential return loss better that 10 dB and a common mode return loss better than 6 dB from 100 MHz to (0.8)*(Baud Frequency). This includes contributions from on-chip circuitry, the chip package and any off-chip components related to the receiver. AC coupling components are included in this requirement. The reference impedance for return loss measurements is 100 Ohm resistive for differential return loss and 25 Ohm resistive for common mode.

Notes

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_{IN}	Differential Input Votlage	200	1600	mV p-p	Measured at receiver
J_D	Deterministic Jitter Tolerance	0.37	-	UI p-p	Measured at receiver
J_{DR}	Combined Deterministic and Random Jitter Tolerance	0.55	-	UI p-p	Measured at receiver
J_T	Total Jitter Tolerance(1)	0.65	-	UI p-p	Measured at receiver
S_{MI}	Multiple Input Skew	-	24	ns	Skew at the receiver input between lanes of a multi-lane link
BER	Bit Error Rate		10^{-12}		
UI	Unit Interval	800	800	ps	+/- 100 ppm

Figure 33 Receiver AC Timing Specifications - 1.25 GBaud

Note:

1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the un-shaded region of Figure 35. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_{IN}	Differential Input Votlage	200	1600	mV p-p	Measured at receiver
J_D	Deterministic Jitter Tolerance	0.37	-	UI p-p	Measured at receiver
J_{DR}	Combined Deterministic and Random Jitter Tolerance	0.55	-	UI p-p	Measured at receiver
J_T	Total Jitter Tolerance(1)	0.65	-	UI p-p	Measured at receiver
S_{MI}	Multiple Input Skew	-	24	ns	Skew at the receiver input between lanes of a multi-lane link
BER	Bit Error Rate		10^{-12}		
UI	Unit Interval	400	400	ps	+/- 100 ppm

Figure 34 Receiver AC Timing Specifications - 2.5 GBaud

Note:

1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the un-shaded region of Figure 35. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects.

Notes

Symbol	Parameter	Range		Unit	Notes
		Min	Max		
V_{IN}	Differential Input Votlage	200	1600	mV p-p	Measured at receiver
J_D	Deterministic Jitter Tolerance	0.37	-	UI p-p	Measured at receiver
J_{DR}	Combined Deterministic and Random Jitter Tolerance	0.55	-	UI p-p	Measured at receiver
J_T	Total Jitter Tolerance(1)	0.65	-	UI p-p	Measured at receiver
S_{MI}	Multiple Input Skew	-	22	ns	Skew at the receiver input between lanes of a multi-lane link
BER	Bit Error Rate		10^{-12}		
UI	Unit Interval	320	320	ps	+/- 100 ppm

Figure 35 Receiver AC Timing Specifications - 3.125 GBaud

Note:

1. Total jitter is composed of three components, deterministic jitter, random jitter and single frequency sinusoidal jitter. The sinusoidal jitter may have any amplitude and frequency in the un-shaded region of Figure 35. The sinusoidal jitter component is included to ensure margin for low frequency jitter, wander, noise, crosstalk and other variable system effects.

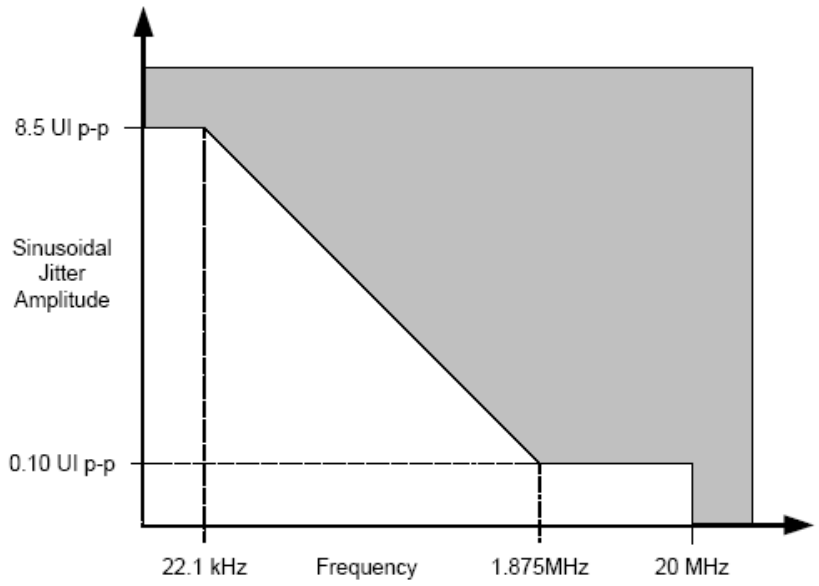


Figure 36 Single Frequency Sinusoidal Jitter Limits

Notes

13.6.1 Receiver Eye Diagrams

For each baud rate at which an LP-Serial receiver is specified to operate, the receiver meets the corresponding Bit Error Rate specification (Receiver AC Timing Specification - 1.25 GBaud, Receiver AC Timing Specification - 2.5 GBaud, and Receiver AC Timing Specification - 3.125 GBaud) when the eye pattern of the receiver test signal (exclusive of sinusoidal jitter) falls entirely within the un-shaded portion of the Receiver Input Compliance Mask shown in Figure 36 with the parameters specified in Figure 37. The eye pattern of the receiver test signal is measured at the input pins of the receiving device with the device replaced with a 100 Ohm +/- 5% differential resistive load.

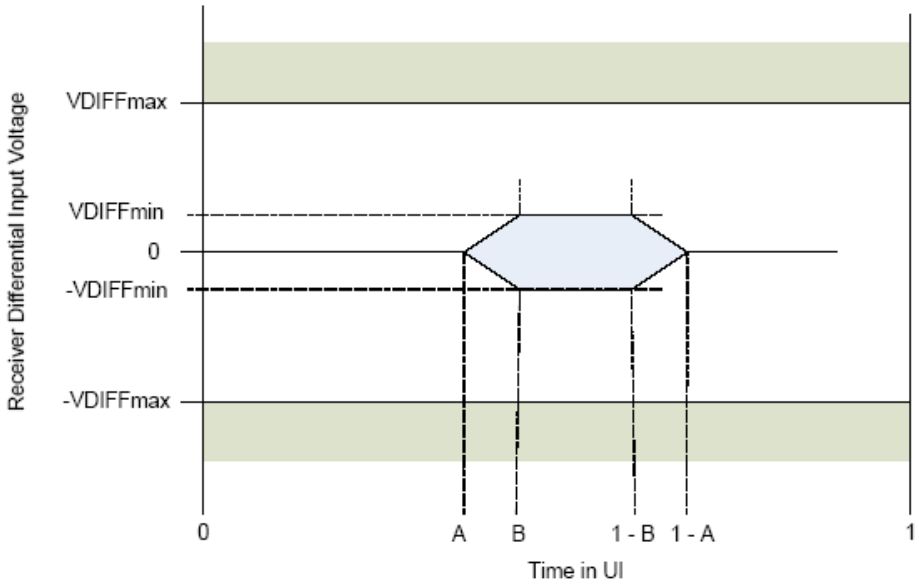


Figure 37 Receiver Input Compliance Mask

Receiver Rate	$V_{DIFFmin}$ (mV)	$V_{DIFFmax}$ (mV)	A (UI)	B (UI)
1.25 GBaud	100	800	0.275	0.400
2.5 GBaud	100	800	0.275	0.400
3.125 GBaud	100	800	0.275	0.400

Figure 38 Receiver Input Compliance Mask Parameters exclusive of Sinusoidal Jitter

Notes

14.0 Parallel Port Electrical Characteristics

The parallel port on the SerB can connect to a QDRII-B4 x36 SRAM. The SerB acts as a memory controller and drives the external SRAM. The P-Port may also be disabled.

The electrical requirements of the P-Port are simply must be QDRII compatible. As a FIFO controller, the SerB must be Burst 4 compatible. Included in the Parallel Port Requirements is the need for programmable output impedance as is used in the QDRII SRAM. This includes the attachment of an external resistor to set the impedance.

The Serial interface operate at 3.125G bps with 8B/10B encoding on each lane. After decoding and alignment of the four lanes, the maximum data rate is 10G bps across the interface in each direction. The external memories are all burst of four. The clock rate on the bus is specified at 156.25 MHz. The 156.25 MHz is sufficient to support the 10G bps total bandwidth in each direction necessary on the P-Port.

Please refer to figure below for SerB to external QDRII SRAM interface connections.

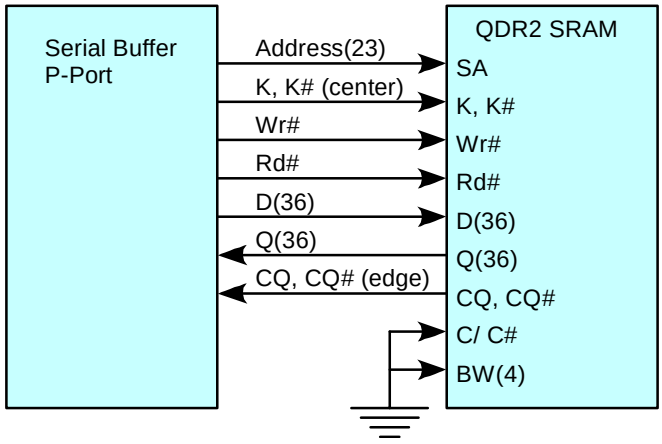


Figure 39 P-Port Signals Connected to a QDRII SRAM

14.1 AC Electrical Characteristics

In this mode, the P-Port electrical characteristics and interface shall be fully compliant with designated QDRII SRAM devices at HSTL levels. While QDRII has the ability to operate at 1.8V and other voltages in between 1.8V and HSTL, there is no requirement for the SerB to operate beyond HSTL. There will be a direct connection from the P-Port to the memory. The drive requirements of the interface will be HSTL Class 1 or less. There is a ZQ pin for setting interface impedance.

When connected to a QDR memory, the specific needs of the QDRII device must be met. The P-Port to QDR SRAM clocking include the following:

- The P-Port output clock / QDR input clock shall be center aligned and designed to clock the QDRII SRAM K/K# input clock.
- The P-Port input clock shall be edge aligned and designed to connect to the CQ/CQ# output of the QDRII SRAM. It is strongly suggested that the C/C# clocks not be returned to the SerB P-Port.
- The SerBs PHY clock is used internally to generate the P-Port output clock

Please refer to table below for specific AC Electrical Characteristics requirements.

Notes

		156.25MHz			
Symbol	**Parameter**	**Min.**	**Max.**	**Unit**	**Notes**
Clock Parameters					
tKHKH	Clock Cycle Time (K,$\overline{K}$,C,$\overline{C}$)	6.00	8.40	ns	
tKC var	Clock Phase Jitter (K,$\overline{K}$,C,$\overline{C}$)	-	0.20	ns	1
tKHKL	Clock High Time (K,$\overline{K}$,C,$\overline{C}$)	2.40	-	ns	5
tKLKH	Clock Low Time (K,$\overline{K}$,C,$\overline{C}$)	2.40	-	ns	5
tKH$\overline{K}$H	Clock to $\overline{clock}$ (K $\overline{K}$,C $\overline{C}$)	2.70	-	ns	6
t$\overline{K}$HKH	$\overline{Clock}$ to clock ($\overline{K}$ K,$\overline{C}$ C)	2.70	-	ns	6
tKHCH	Clock to data clock (K C,$\overline{K}$ $\overline{C}$)	0.00	2.80	ns	
tCK lock	DLL lock tim (K, C)	1024	-	cycles	2
tKC reset	K static to DLL reset	30	-	ns	
Output Parameters					
tCHQV	C, $\overline{C}$ HIGH to output valid	-	0.50	ns	3
tCHQX	C, $\overline{C}$ HIGH to output hold	-0.50	-	ns	3
tCHCQV	C, $\overline{C}$ HIGH to echo clock valid	-	0.50	ns	3
tCHCQX	C, $\overline{C}$ HIGH to echo clock hold	-0.50	-	ns	3
tCQHQV	CQ, $\overline{CQ}$ HIGH to output valid	-	0.40	ns	
tCQHQX	CQ, $\overline{CQ}$ HIGH to output hold	-0.40	-	ns	
tCHQZ	$\overline{C}$ HIGH to output High-Z	-	0.50	ns	3
tCHQX1	$\overline{C}$ HIGH to output Low-Z	-0.50	-	ns	3
Set-Up Times					
tAVKH	Address valid to K, $\overline{K}$ rising edge	0.50	-	ns	4
tIVKH	$\overline{R}$, $\overline{W}$ inputs valid to K, $\overline{K}$ rising edge	0.50	-	ns	
tDVKH	Data-in valid to K, $\overline{K}$ rising edge	0.50	-	ns	
Hold Times					
tKHAX	K, $\overline{K}$ rising edge to address hold	0.50	-	ns	6
tKHIX	K, $\overline{K}$ rising edge to $\overline{R}$, $\overline{W}$ inputs hold	0.50	-	ns	
tKHDX	K, $\overline{K}$ rising edge to data-in hold	0.50	-	ns	

Table 102 AC Electrical Characteristics

Note:

1. Clock phase jitter is the variance from clock rising edge to the next expected clock rising edge.
2. Vdd slew rate must be less than 0.1V DC per 50ns for DLL lock retention. DLL lock time begins once Vdd and input clock are stable.
3. If C, $\overline{C}$ are tied High, K, $\overline{K}$ become the references for C, $\overline{C}$ timing parameters.
4. All address inputs must meet the specified setup and hold times for all latching clock edges.
5. Clock High time (tKHKL) and Clock Low time (tKLKH) should be within 40% to 60% of the duty cycle time (tKHKH).
6. Clock to $\overline{Clock}$ time (tKH$\overline{K}$H) and $\overline{Clock}$ to Clock time (t$\overline{K}$HKH) should be within 45% to 55% of the duty cycle time (tKHKH).

Notes

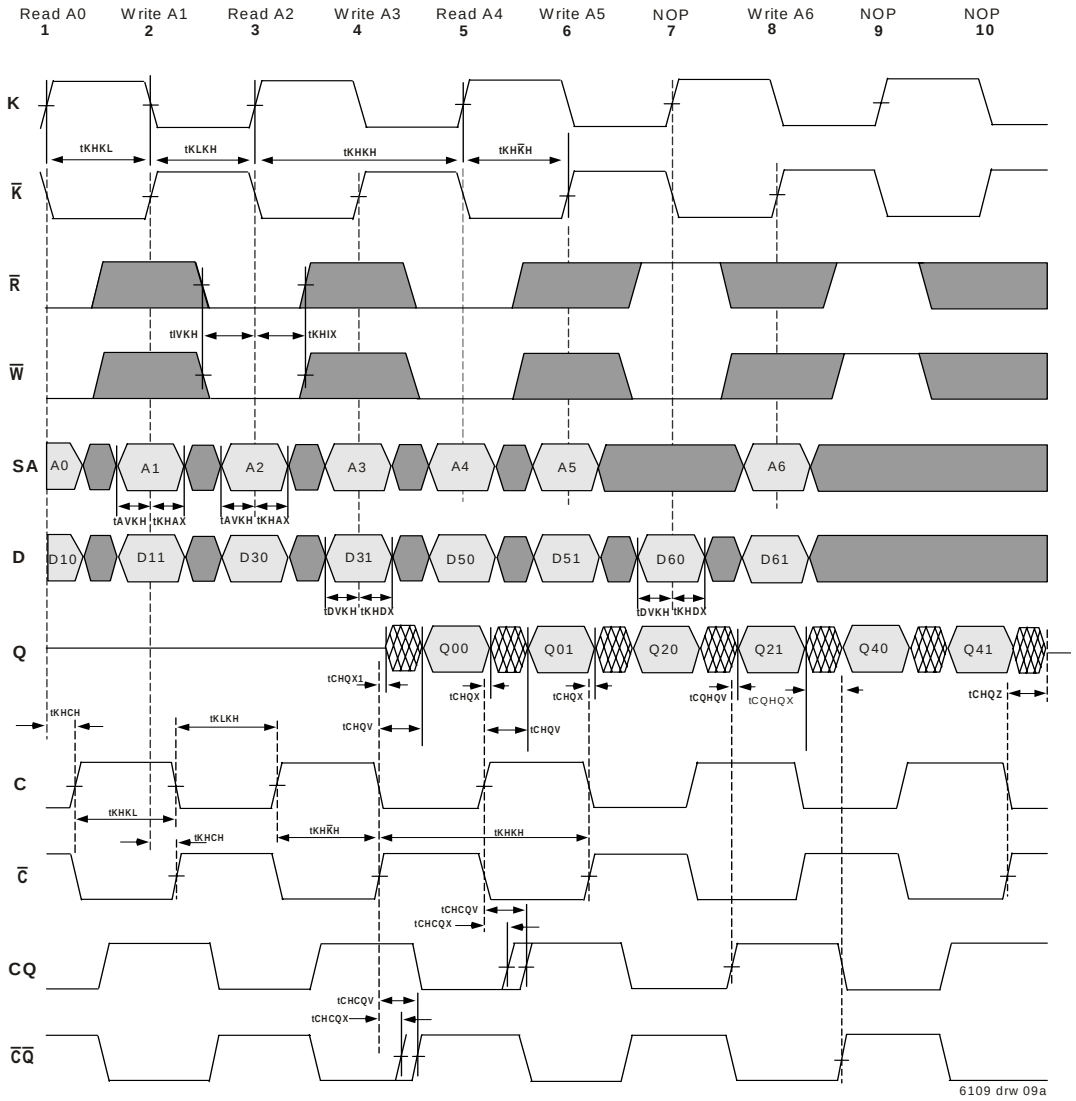


Figure 40 Timing Waveform of Combined Read and Write Cycles

Notes

Notes

15.0 JTAG Interface

The 80KSBR200 offers full JTAG (Boundary Scan) support for both its slow speed and high speed pins. This allows “pins-down” testing of newly manufactured printed circuit boards as well as troubleshooting of field returns. The JTAG TAP interface also offers an alternative method for Configuration Register Access (CRA) (along with the sRIO and I^2C ports). Thus this port may be used for programming the SerB’s many registers.

Boundary scan testing of the AC-coupled IOs is performed in accordance with IEEE 1149.6 (AC Extest).

15.1 IEEE 1149.1 (JTAG) & IEEE 1149.6 (AC Extest) Compliance

All DC pins are in full compliance with IEEE 1149.1 [10]. All AC-coupled pins fully comply with IEEE 1149.6 [11]. All 1149.1 and 1149.6 boundary scan cells are on the same chain. No additional control cells are provided for independent selection of negative and/or positive terminals of the TX- or RX-pairs.

15.2 System Logic TAP Controller Overview

The system logic utilizes a 16-state, six-bit TAP controller, a four-bit instruction register, and five dedicated pins to perform a variety of functions. The primary use of the JTAG TAP Controller state machine is to allow the five external JTAG control pins to control and access the SerB's many external signal pins. The JTAG TAP Controller can also be used for identifying the device part number. The JTAG logic of the 80KSBR200 is depicted in the figure below.

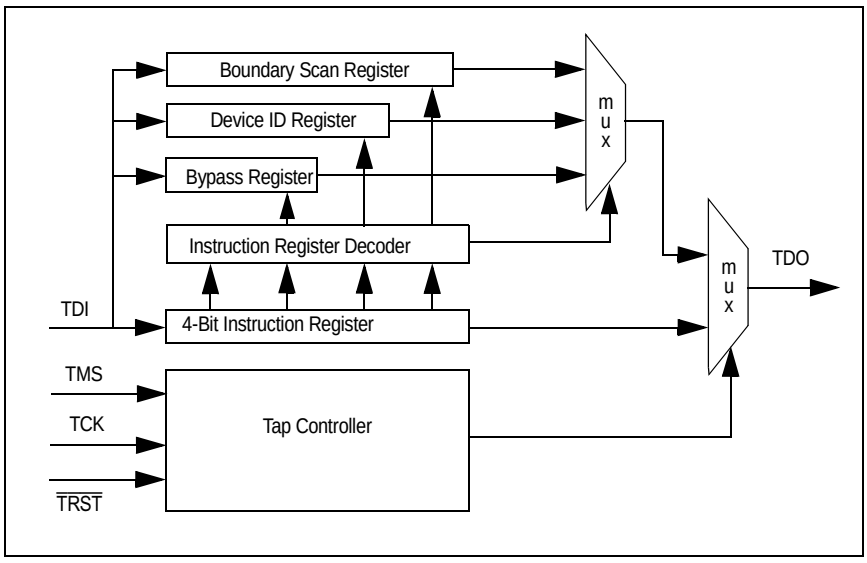


Figure 41 Diagram of the JTAG Logic

15.3 Signal Definitions

JTAG operations such as Reset, State-transition control and Clock sampling are handled through the signals listed in the table below. A functional overview of the TAP Controller and Boundary Scan registers is provided in the sections following the table.

Pin Name	Type	Description
$\overline{\text{TRST}}$	Input	JTAG RESET Asynchronous reset for JTAG TAP controller (internal pull-up)
TCK	Input	JTAG Clock Test logic clock. JTAG_TMS and JTAG_TDI are sampled on the rising edge. JTAG_TDO is output on the falling edge.
TMS	Input	JTAG Mode Select. Requires an external pull-up. Controls the state transitions for the TAP controller state machine (internal pull-up)

Table 103 JTAG Pin Descriptions (Part 1 of 2)

Notes

Pin Name	Type	Description
TDI	Input	JTAG Input Serial data input for BSC chain, Instruction Register, IDCODE register, and BYPASS register (internal pull-up)
TDO	Output	JTAG Output Serial data out. Tri-stated except when shifting while in Shift-DR and SHIFT-IR TAP controller states.

Table 103 JTAG Pin Descriptions (Part 2 of 2)

The system logic TAP controller transitions from state to state, according to the value present on JTMS, as sampled on the rising edge of TCK. The Test-Logic Reset state can be reached either by asserting $\overline{TRST}$ or by applying a 1 to TMS for five consecutive cycles of TCK. A state diagram for the TAP controller appears in Figure 41. The value next to state represent the value that must be applied to TMS on the next rising edge of TCK, to transition in the direction of the associated arrow.

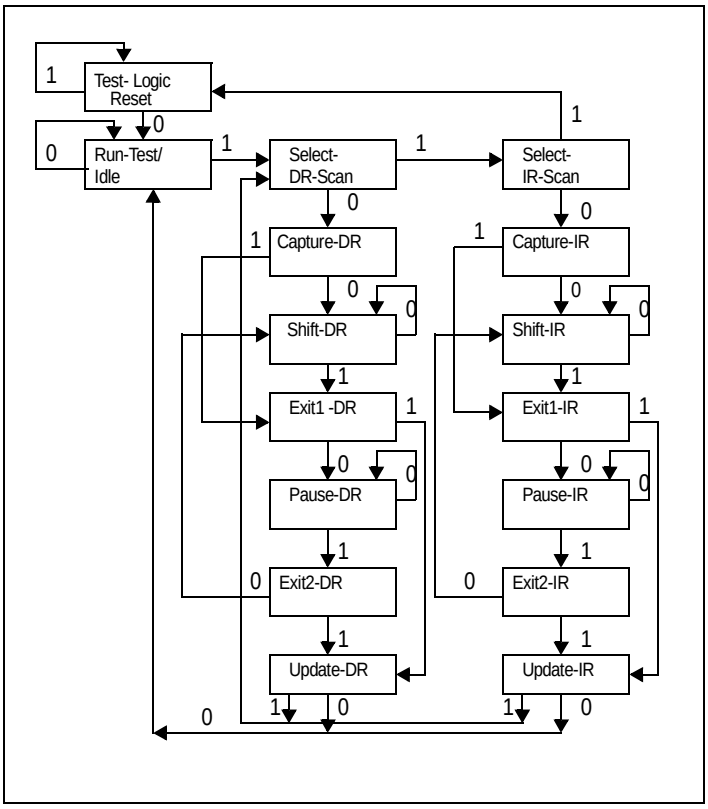


Figure 42 State Diagram of the 80KSBR200's TAP Controller

15.4 Test Data Register (DR)

The Test Data register contains the following:

- The Bypass register
- The Boundary Scan registers
- The Device ID register

These registers are connected in parallel between a common serial input and a common serial data output, and are described in the following sections. For more detailed descriptions, refer to IEEE Standard Test Access port (IEEE Std. 1149.1-1990).

Notes

15.4.1 Boundary Scan Registers

The 80KSBR200 boundary scan chain is 140 bits long. The five JTAG pins do not have scan elements associated with them. Full boundary scan details can be found in the associated BSDL file which may be found on our web site (www.IDT.com). The boundary scan chain is connected between TDI and TDO when the EXTEST or SAMPLE/PRELOAD instructions are selected. Once EXTEST is selected and the TAP controller passes through the UPDATE-IR state, whatever value that is currently held in the boundary scan register's output latches is immediately transferred to the corresponding outputs or output enables.

Therefore, the SAMPLE/PRELOAD instruction must first be used to load suitable values into the boundary scan cells, so that inappropriate values are not driven out onto the system pins. All of the boundary scan cells feature a negative edge latch, which guarantees that clock skew cannot cause incorrect data to be latched into a cell. The input cells are sample-only cells.

The simplified logic configuration is shown in the figure below.

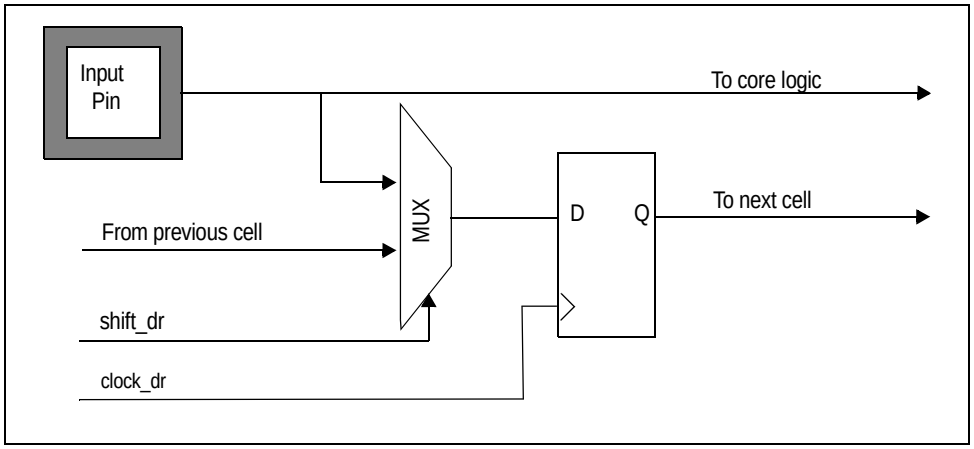


Figure 43 Diagram of Observe-only Input Cell

The simplified logic configuration of the output cells is shown in the figure below.

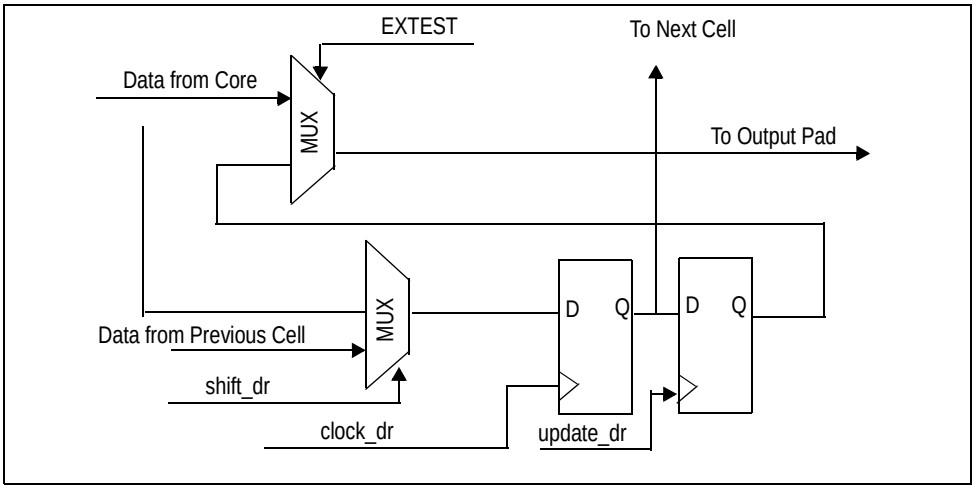


Figure 44 Diagram of Output Cell

Notes

The output enable cells are also output cells. The simplified logic appears in the figure below.

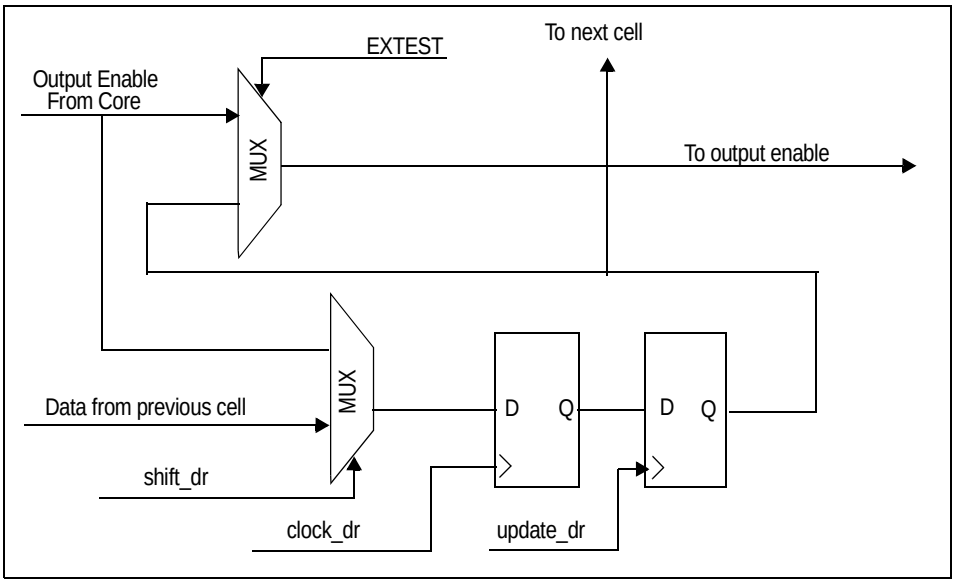


Figure 45 Diagram of Output Enable Cell

The bidirectional cells are composed of only two boundary scan cells. They contain one output enable cell and one capture cell, which contains only one register. The input to this single register is selected via a mux that is selected by the output enable cell when EXTEST is disabled. When the Output Enable Cell is driving a high out to the pad (which enables the pad for output) and EXTEST is disabled, the Capture Cell will be configured to capture output data from the core to the pad.

However, in the case where the Output Enable Cell is low (signifying a tri-state condition at the pad) or EXTEST is enabled, the Capture Cell will capture input data from the pad to the core. The configuration is shown graphically in the figure below.

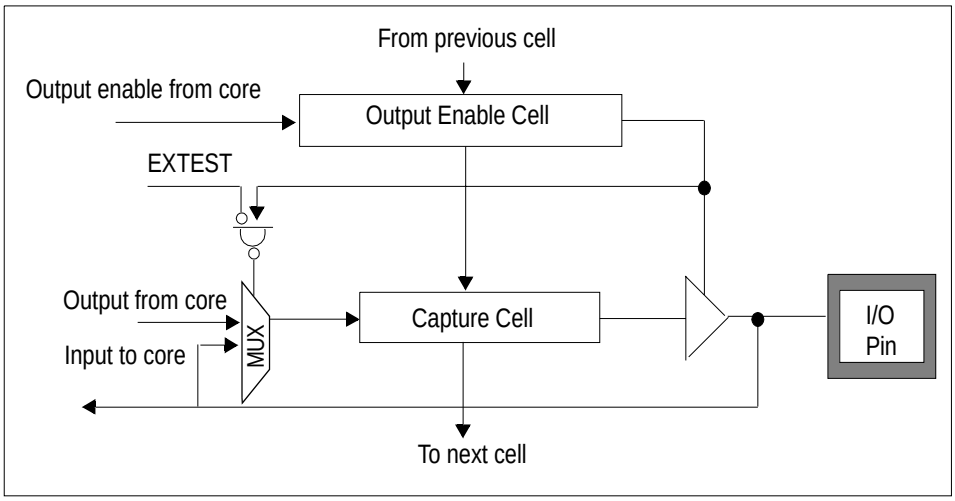


Figure 46 Diagram of Bidirectional Cell

Notes

15.5 Instruction Register (IR)

The Instruction register allows an instruction to be shifted serially into the SerB at the rising edge of TCK. The instruction is then used to select the test to be performed or the test register to be accessed, or both. The instruction shifted into the register is latched at the completion of the shifting process, when the TAP controller is at the Update-IR state.

The Instruction Register contains four shift-register-based cells that can hold instruction data. This register is decoded to perform the following functions:

- To select test data registers that may operate while the instruction is current. The other test data registers should not interfere with chip operation and selected data registers.
- To define the serial test data register path used to shift data between TDI and TDO during data register scanning.

The Instruction Register is comprised of 4 bits to decode instructions, as shown in the table below.

Instruction	Definition	OPcode [3:0]
EXTEST	Mandatory instruction allowing the testing of board level interconnections. Data is typically loaded onto the latched parallel outputs of the boundary scan shift register using the SAMPLE/PRELOAD instruction prior to use of the EXTEST instruction. EXTEST will then hold these values on the outputs while being executed. Also see the CLAMP instruction for similar capability.	0000
SAMPLE/ PRELOAD	Mandatory instruction that allows data values to be loaded onto the latched parallel output of the boundary-scan shift register prior to selection of the other boundary-scan test instruction. The Sample instruction allows a snapshot of data flowing from the system pins to the on-chip logic or vice versa.	0001
IDCODE	Provided to select Device Identification to read out manufacturer's identity, part, and version number.	0010
HIGHZ	Tri-states all output and bidirectional boundary scan cells.	0011
CLAMP	Provides JTAG user the option to bypass the part's JTAG controller while keeping the part outputs controlled similar to EXTEST.	0100
EXTEST_PULSE	AC Extest instruction implemented in accordance with the requirements of the IEEE std. 1149.6 specification.	0101
EXTEST_TRAIN	AC Extest instruction implemented in accordance with the requirements of the IEEE std. 1149.6 specification.	0110
RESERVED	Behaviorally equivalent to the BYPASS instruction as per the IEEE std. 1149.1 specification. However, the user is advised to use the explicit BYPASS instruction.	0111 — 1001
CONFIGURATION REGISTER ACCESS (CRA)	SerB-specific opcode to allow reading and writing of the configuration registers. Reads and writes must be 32-bits. See further detail below.	1010
PRIVATE	For internal use only. Do not use.	1011 — 1100
SHIFT FUSE STATUS	To shift the internal fuse status out to TDO pin.	1101
PRIVATE	For internal use only. Do not use.	1110
BYPASS	The BYPASS instruction is used to truncate the boundary scan register as a single bit in length.	1111

Table 104 Instructions Supported By 80KSBR200's JTAG Boundary Scan

15.5.1 EXTEST

The external test (EXTEST) instruction is used to control the boundary scan register, once it has been initialized using the SAMPLE/PRELOAD instruction. Using EXTEST, the user can then sample inputs from or load values onto the external pins of the 80KSBR200. Once this instruction is selected, the user then uses the SHIFT-DR TAP controller state to shift values into the boundary scan chain. When the TAP controller passes through the UPDATE-DR state, these values will be latched onto the output pins or into the output enables.

Notes

15.5.2 SAMPLE/PRELOAD

The sample/preload instruction has a dual use. The primary use of this instruction is for pre-loading the boundary scan register prior to enabling the EXTEST instruction. Failure to preload will result in unknown random data being driven onto the output pins when EXTEST is selected. The secondary function of SAMPLE/PRELOAD is for sampling the system state at a particular moment. Using the SAMPLE function, the user can halt the device at a certain state and shift out the status of all of the pins and output enables at that time.

15.5.3 BYPASS

The BYPASS instruction is used to truncate the boundary scan register to a single bit in length. During system level use of the JTAG, the boundary scan chains of all the devices on the board are connected in series. In order to facilitate rapid testing of a given device, all other devices are put into BYPASS mode. Therefore, instead of having to shift 140 times to get a value through the 80KSBR200, the user only needs to shift one time to get the value from TDI to TDO. When the TAP controller passes through the CAPTURE-DR state, the value in the BYPASS register is updated to be 0.

If the device being used does not have an IDCODE register, then the BYPASS instruction will automatically be selected into the instruction register whenever the TAP controller is reset. Therefore, the first value that will be shifted out of a device without an IDCODE register is always 0. Devices such as the 80KSBR200 that include an IDCODE register will automatically load the IDCODE instruction when the TAP controller is reset, and they will shift out an initial value of 1. This is done to allow the user to easily distinguish between devices having IDCODE registers and those that do not.

15.5.4 CLAMP

This instruction, listed as optional in the IEEE 1149.1 JTAG Specifications, allows the boundary scan chain outputs to be clamped to fixed values. When the clamp instruction is issued, the scan chain will bypass the 80KSBR200 and pass through to devices further down the scan chain.

15.5.5 IDCODE

The IDCODE instruction is automatically loaded when the TAP controller state machine is reset either by the use of the $\overline{\text{TRST}}$ signal or by the application of a '1' on TMS for five or more cycles of TCK as per the IEEE Std 1149.1 specification. The least significant bit of this value must always be 1. Therefore, if a device has a IDCODE register, it will shift out a 1 on the first shift if it is brought directly to the SHIFT-DR TAP controller state after the TAP controller is reset. The board- level tester can then examine this bit and determine if the device contains a DEVICE_ID register (the first bit is a 1), or if the device only contains a BYPASS register (the first bit is 0).

However, even if the device contains an IDCODE register, it must also contain a BYPASS register. The only difference is that the BYPASS register will not be the default register selected during the TAP controller reset. When the IDCODE instruction is active and the TAP controller is in the Shift-DR state, the thirty-two bit value that will be shifted out of the device-ID register is `0x004F0037`.

Bit(s)	Mnemonic	Description	R/W	Reset
0	reserved	reserved 0x1	R	1
11:1	Manuf_ID	**Manufacturer Identity** (11 bits) IDT 0x33	R	0x033
27:12	Part_number	**Part Number** (16 bits) This field identifies the part number of the processor derivative. For the 80KSBR200 this value is: 0x04F0	R	impl. dep.
31:28	Version	**Version** (4 bits) This field identifies the version number of the processor derivative. For the 80KSBR200, this value is 0x0	R	impl. dep.

Table 105 System Controller Device Identification Register

Notes

Version	Part Number	Manuf ID	LSB
0000	0000|0100|1111|0000	0000|0011|011	1

Figure 106 System Controller Device ID Instruction Format

15.5.6 EXTEST PULSE

This IEEE 1149.6 instruction applies only to the AC-coupled pins. All DC pins will perform as if the IEEE Std 1149.1 EXTEST instruction is operating whenever the EXTEST_PULSE instruction is effective.

The EXTEST_PULSE instruction enables edge-detecting behavior on signal paths containing AC pins, where test receivers reconstruct the original waveform created by a driver even when signals decay due to AC-coupling.

As the operation name suggests, enabling EXTEST_PULSE causes a pulse to be issued which can be detected even on AC-coupled receivers. Refer to the IEEE Std 1149.6 for full details. Below is a short synopsis.

If enabled, the output signal is forced to the value in its associated Boundary-Scan Register data cell for its driver (true and inverted values for a differential pair) at the falling edge of TCK in the Update-IR and Update-DR TAP Controller states. The output subsequently transitions to the opposite of that state (an inverted state) on the first falling edge of TCK that occurs after entering the Run-Test/Idle TAP Controller state. It then transitions back again to the original state (a non-inverted state) on the first falling edge of TCK after leaving the Run-Test/Idle TAP Controller state.

15.5.7 EXTEST TRAIN

This IEEE 1149.6 instruction applies only to the AC-coupled pins. All DC pins will perform as if the IEEE Std 1149.1 EXTEST instruction is operating whenever the EXTEST_PULSE instruction is effective.

The EXTEST_TRAIN instruction enables edge-detecting behavior on signal paths containing AC pins, where test receivers reconstruct the original waveform created by a driver even when signals decay due to AC-coupling.

As the operation name suggests, enabling EXTEST_TRAIN causes a pulse train to be issued which can be detected even on AC-coupled receivers. Once in an enabled state, the train will be sent continuously in response to the TCK clock. No other signaling is required to generate the pulse train while in this state. Refer to the IEEE Std 1149.6 for full details. Below is a short synopsis.

First, the output signal is forced to the state matching the value (a non-inverted state) in its associated Boundary-Scan Register data cell for its driver (true and inverted values for a differential pair), at the falling edge of TCK in update-IR. Then the output signal transitions to the opposite state (an inverted state) on the first falling edge of TCK that occurs after entering the Run-Test/Idle TAP Controller state. While remaining in this state, the output signal will continue to invert on every falling edge of TCK, thereby generating a pulse train.

15.5.8 RESERVED

Reserved instructions implement various test modes used in the device manufacturing process. The user should not enable these instructions.

15.5.9 PRIVATE

Private instructions implement various test modes used in the device manufacturing process. The user should not enable these instructions.

15.6 Usage Considerations

As previously stated, there are internal pull-ups on $\overline{\text{TRST}}$, TMS, and TDI. However, TCK also needs to be driven to a known value. It is best to either drive a zero on the TCK pin when it is not being used or to use an external pull-down resistor. In order to guarantee that the JTAG does not interfere with normal system operation, the TAP controller should be forced into the Test-Logic-Reset controller state by continuously holding $\overline{\text{TRST}}$ low and/or TMS high when the chip is in normal operation. If JTAG will not be used, externally pull-down $\overline{\text{TRST}}$ low to disable it.

Notes

15.7 JTAG Configuration Register Access

As previously mentioned, the JTAG port may be used to read and write to the 80KSBR200's configuration registers. The same JTAG instruction (4b1010) is used for both writes and reads.

Bits	Field Name	Size	Description
0	jtag_config_wr_n	1	1 – read configuration register 0 – write configuration register
[22:1]	jtag_config_addr	22	Starting address of the memory mapped configuration register. 22 address bits map to a unique double-word aligned on a 32-bit boundary. This provides accessibility to and is consistent with the sRIO memory mapping.
[54:23]	jtag_config_data	32	Reads: Data shifted out (one 32-bit word per read) is read from the configuration register at address jtag_config_addr. Writes: Data shifted in (one 32-bit word per write) is written to the configuration register at address jtag_config_addr.

Table 107 Data Stream for JTAG Configuration Register Access Mode

15.7.1 Writes during Configuration Register Access

A write is performed by shifting the CRA OPcode into the Instruction Register (IR), then shifting in first a read / write select bit, then both the 22-bit target address and 32-bit data into the Data Register (DR). When bit 0 of the data stream is 0, data shifted in after the address will be written to the address specified in jtag_config_addr. The TDO pin will transmit all 0s. See the figure below for the associated timing diagram.

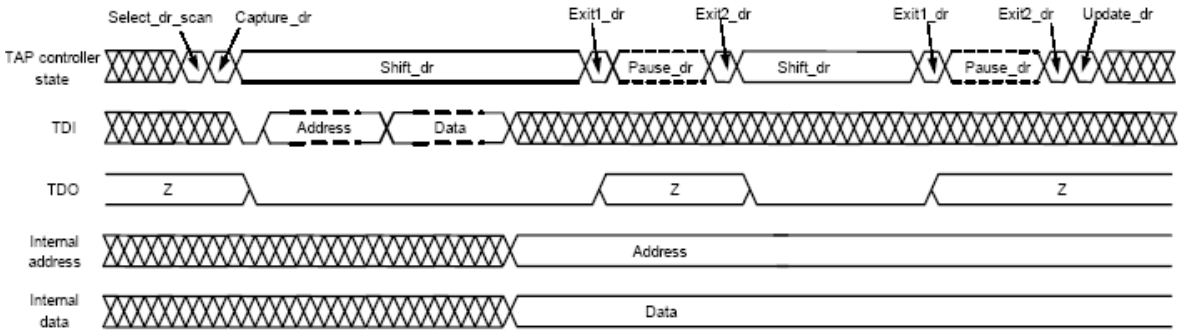


Figure 47 Implementation of write during configuration register access

15.7.2 Reads during Configuration Register Access

Reads are much like writes except that target data is not provided. When bit 0 of the data stream is 1, data shifted out will be read from the address specified in jtag_config_addr. TDI will not be used after the address is shifted in. As a function of read latency in the architecture, the first 16 bits will be 0's and must be ignored. The following bits will contain the actual register bits.

Notes

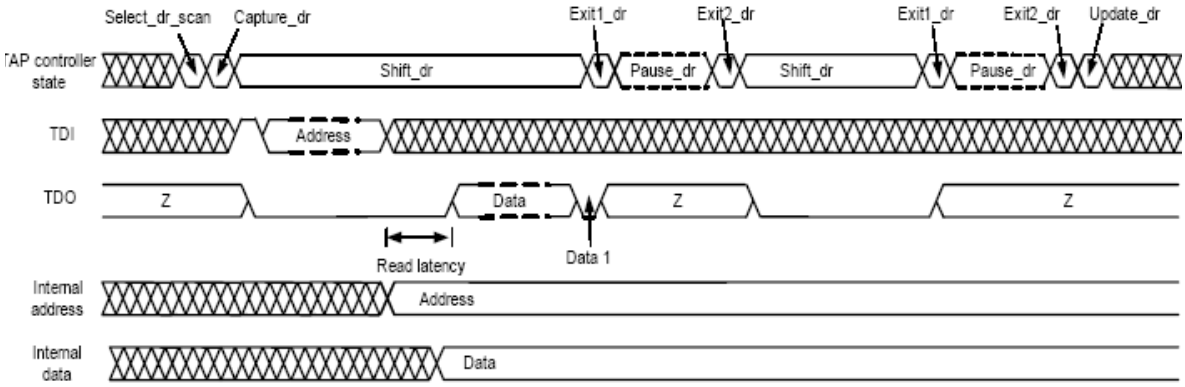


Figure 48 Implementation of read during configuration register access

15.8 JTAG DC Electrical Specifications

At recommended operating conditions with V_{DD3} = 3.3V ± 5%

Parameter	Symbol	Min	Max	Unit
3.3V Supply Voltage	V_{DD3}	3.15	3.45	V
Ground	V_{SS}	0	0	V
Input high voltage level	V_{IH}	2.0	$V_{DD3(max)}$ + 0.15	V
Input low voltage level	V_{IL}	-0.3	0.8	V
Output Low Voltage (I_{OL} = 4mA, V_{DD3} = Min.)	V_{OL}	-	0.4	V
Output High Voltage (IOH = -4mA, V_{DD3} = Min.)	V_{OH}	2.4	-	V
Input current for JTAG pins (input voltage is between 0.1 x V_{DD3} and 0.9 x V_{DD3} (max))	I_{LI}	-30	30	uA
Capacitance for each Input pin	C_{IN}	-	8	pF
Capacitance for each I/O or Output pin	C_{OUT}	-	10	pF

Figure 49 JTAG DC Electrical Specifications (V_{DD3} = 3.3V ± 5%)

At recommended operating conditions with V_{DD3} = 2.5V ± 100mV

Parameter	Symbol	Min	Max	Unit
2.5V Supply Voltage	V_{DD3}	2.4	2.6	V
Ground	V_{SS}	0	0	V
Input high voltage level	V_{IH}	1.7	$V_{DD3(max)}$ + 0.1	V
Input low voltage level	V_{IL}	-0.3	0.7	V
Output Low Voltage (I_{OL} = 2mA, V_{DD3} = Min.)	V_{OL}	-	0.4	V
Output High Voltage (I_{OH} = -2mA, V_{DD3} = Min.)	V_{OH}	2.0	-	V
Input current for JTAG pins (input voltage is between 0.1 x V_{DD3} and 0.9 x V_{DD3} (max)	I_{LI}	-30	30	uA
Capacitance for each Input pin	C_{IN}	-	8	pF
Capacitance for each I/O or Output pin	C_{OUT}	-	10	pF

Figure 50 JTAG DC Electrial Specifications (V_{DD3} = 2.5V ± 100mV)

Notes

15.9 JTAG AC Electrical Specifications

Symbol	Parameter	80KSBR200		
		Min.	Max.	Units
t_{JCYC}	JTAG Clock Input Period	100	—	ns
t_{JCH}	JTAG Clock HIGH	40	—	ns
t_{JCL}	JTAG Clock Low	40	—	ns
t_{JR}	JTAG Clock Rise Time	—	3(1)	ns
t_{JF}	JTAG Clock Fall Time	—	3(1)	ns
t_{JRST}	JTAG Reset	50	—	ns
t_{JRSR}	JTAG Reset Recovery	50	—	ns
t_{JCD}	JTAG Data Output	—	25	ns
t_{JDC}	JTAG Data Output Hold	0	—	ns
t_{JS}	JTAG Setup	15	—	ns
t_{JH}	JTAG Hold	15	—	ns

5686tbl 02

Figure 51 JTAG AC Electrical Specifications

Note:

1. Guaranteed by design.
2. 30pF loading on external output signals.
3. Refer to AC Electrical Test Conditions stated earlier in this document.
4. JTAG operations occur at one speed (10MHz). The base device may run at any speed specified in this datasheet.

15.10 JTAG Timing Specification

Symbol	Parameter	Range		Unit	Notes
		Min.	Max.		
V_O	Output Voltage	-0.40	2.30	Volts	Voltage relative to COMMON of either signal comprising a differential pair
$V_{DIFF\ PP}$	Differential Output Voltage	800	1600	mV p-p	
J_D	Deterministic Jitter	—	0.17	UI p-p	
J_T	Total Jitter	—	0.35	UI p-p	
S_{MO}	Multiple Output Skew	—	1000	ps	Skew at the transmitter output between lanes of a multilane link
UI	Unit Interval	320	320	ps	+/- 100 ppm

5686 tbl 08

Figure 52 JTAG Timing Specifications

Note:

1. Device inputs = All device inputs except TDI, TMS, and TRST.
2. Device outputs = All device outputs except TDO.

16.0 Pinout & Pin Listing

16.1 Pinout

TOP VIEW

index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A	GND	VDD3	TMS	TRST	TDO	ID0	ID2	GND	GND	GND	MBDONE	TM2	VREF	TM0	D22	D1	D7	D15	D27	D35	VDDQ	GND
B	VDD3	GND	MRST_N	TCK	TDI	ID1	ID3	GND	GND	PPE_N	MBPASS	SCEN	TM1	D8	D19	D4	D12	D24	D33	D26	D23	D11
C	GND	VDD3	GND	SCL	SDA	ADS	GNDS	ID5	GNDS	ID7	ID8	ID9	D0	D5	D16	VDDQ	D18	GND	D32	VDDQ	D14	GND
D	GND	VDD	GND	VDD	GNDS	ID4	VDDS	ID6	VDDS	GND	VDD	GND	VDD	D2	D13	D25	D21	D30	D29	D20	D9	D6
E	VDDS	GNDS	VDDS	GNDS	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	D10	D28	D31	D34	D17	GND	CKI	CKI_N
F	S1 TXN2	S1 TXP2	VDDA	S1 TXN3	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDD	VDDQ	GND	D3	GND	VDDQ
G	GNDA	VDDA	GNDA	S1 TXP3	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	CKO_N	Q35	VDDQ	Q28	Q29
H	S1 RXP2	GNDA	VDDA	GNDS	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDDQ	CKO	Q32	GND	Q26	Q27
J	S1 RXN2	VDDS	S1 RXP3	S1 RXN3	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	Q34	Q33	VDDQ	Q22	Q23
K	VDDS	GNDS	VDDS	GNDS	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDDQ	Q30	Q31	GND	Q20	Q21
L	S1 RXN0	VDDS	S1 RXP1	S1 RXN1	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	Q24	Q25	VDDQ	Q18	Q19
M	S1 RXP0	GNDA	VDDA	GNDS	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDDQ	DNC	DNC	GND	Q17	Q16
N	GNDA	VDDA	GNDA	S1 TXP1	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	Q9	Q8	VDDQ	Q15	Q14
P	S1 TXN0	S1 TXP0	VDDA	S1 TXN1	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDDQ	Q0	Q2	GND	Q13	Q12
R	VDDS	GNDS	VDDS	GNDS	VDDS	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	Q1	Q3	VDDQ	Q11	Q10
T	GNDS	VDDS	GNDS	VDDS	GNDS	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	VDDQ	DNC	DNC	GND	Q4	Q6
U	REXTP	IDS	AUX CKI	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDD	GND	VDDQ	GND	A20	WRO_N	VDDQ	Q5	Q7
V	REXTN	GNDS	AUX CKQ	GNDS	VDDS	GNDS	VDDS	GNDS	VDDS	GNDS	VDDS	GNDS	VDDS	GNDS	VDDS	GNDS	A0	A18	A12	GND	RDO_N	A19
W	GND	GNDS	SP1S0	VDDS	GNDS	S2 RXN1	S2 RXP1	GNDS	S2 TXN1	GNDS	S2 TXN3	GNDS	S2 RXP3	S2 RXN3	GNDS	VDDS	A2	A8	A10	VDDQ	A16	A17
Y	IRQ1	STOD	SP1S1	GNDS	VDDS	VDDA	GNDA	VDDA	S2 TXP1	VDDS	S2 TXP3	VDDA	GNDA	VDDA	VDDS	GNDS	PLL_OFF	A6	GND	A9	A11	A15
AA	IRQ0	STOA	REF CLKP	VDDS	GNDS	S2 RXP0	VDDA	GNDA	VDDS	GNDS	VDDS	GNDA	VDDA	S2 RXP2	GNDS	VDDS	QDRA	A4	A14	A5	A7	A13
AB	GND	GND	REF CLKN	GNDS	VDDS	S2 RXN0	GNDA	S2 TXP0	S2 TXN0	VDDS	S2 TXN2	S2 TXP2	GNDA	S2 RXN2	VDDS	GND	ZQ	QDRB	A1	A3	VDDQ	GND

Figure 53 80KSBR200 Pinout

16.2 Pin Listing

Table 108 Pin Listing (Alphabetical)

Pin Number	Pin Name	Function	Supply / Interface	Pin Function Description
V17	**A0**	QDR ADDR 0	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AB19	**A1**	QDR ADDR 1	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
W17	**A2**	QDR ADDR 2	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AB20	**A3**	QDR ADDR 3	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AA18	**A4**	QDR ADDR 4	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AA20	**A5**	QDR ADDR 5	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
Y18	**A6**	QDR ADDR 6	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AA21	**A7**	QDR ADDR 7	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
W18	**A8**	QDR ADDR 8	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
Y20	**A9**	QDR ADDR 9	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
W19	**A10**	QDR ADDR 10	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
Y21	**A11**	QDR ADDR 11	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
V19	**A12**	QDR ADDR 12	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
AA22	**A13**	QDR ADDR 13	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.

AA19	**A14**	QDR ADDR 14	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
Y22	**A15**	QDR ADDR 15	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
V21	**A16**	QDR ADDR 16	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
W22	**A17**	QDR ADDR 17	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
V18	**A18**	QDR ADDR 18	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
V22	**A19**	QDR ADDR 19	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
U18	**A20**	QDR ADDR 20	(VDDQ, GND) / CMOS Output	When operating as a FIFO controller, the A output is the address for the external memory and should be connected directly to the SA lines on the QDRII B4 SRAM.
C6	**ADS**	I^2C	(VDD, GND) / CMOS Input	I^2C address width select. Set ADS = GND for 7-bit SerB slave address. ADS = Vdd for 10-bit. NOTE: SUPPLY / LEVELS REQUIREMENTS ARE UNQUE FROM THE OTHER I^2C PINS.
U3	**AUXCKI**	AUX ClockI		Auxiliary clocks provided to bypass CDR block for DC-type testing of SERDES RX inputs.
V3	**AUXCKQ**	AUX ClockQ		Auxiliary clocks provided to bypass CDR block for DC-type testing of SERDES RX inputs.
E21	**CKI**	P-Port Clock	(VDD, GND) / CMOS Input	Clock input for the P-Port. These inputs should be connected to the CQ/nCQ outputs of the QDR SRAM when operating as a FIFO controller.
E22	**CKI_N**	P-Port Clock	(VDD, GND) / CMOS Input	Clock input for the P-Port. These inputs should be connected to the CQ/nCQ outputs of the QDR SRAM when operating as a FIFO controller.
H18	**CKO**	Echo Clock	(VDD, GND) / CMOS Output	Clock output that is closely aligned with parallel port data output (Q), address (A), Queue Empty (E), and Queue Full (F). When operating as a FIFO controller, outputs read (nRd), and write (nWr) are also aligned. The alignment is selectable as either center aligned or edge aligned in the configuration register. When PPM is LOW, this output should be connected to the K and nK inputs of the QDR SRAM.
G18	**CKO_N**	Echo Clock	(VDD, GND) / CMOS Output	Clock output that is closely aligned with parallel port data output (Q), address (A), Queue Empty (E), and Queue Full (F). When operating as a FIFO controller, outputs read (nRd), and write (nWr) are also aligned. The alignment is selectable as either center aligned or edge aligned in the configuration register. When PPM is LOW, this output should be connected to the K and nK inputs of the QDR SRAM.

C13	**D0**	QDR SRAM Data In 0	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 0
A16	**D1**	QDR SRAM Data In 1	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 1
D14	**D2**	QDR SRAM Data In 2	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 2
F20	**D3**	QDR SRAM Data In 3	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 3
B16	**D4**	QDR SRAM Data In 4	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 4
C14	**D5**	QDR SRAM Data In 5	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 5
D22	**D6**	QDR SRAM Data In 6	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 6
A17	**D7**	QDR SRAM Data In 7	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 7
B14	**D8**	QDR SRAM Data In 8	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 8
D21	**D9**	QDR SRAM Data In 9	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 9
E15	**D10**	QDR SRAM Data In 10	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 10
B22	**D11**	QDR SRAM Data In 11	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 11
B17	**D12**	QDR SRAM Data In 12	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 12
D15	**D13**	QDR SRAM Data In 13	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 13
C21	**D14**	QDR SRAM Data In 14	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 14
A18	**D15**	QDR SRAM Data In 15	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 15
C15	**D16**	QDR SRAM Data In 16	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 16
E19	**D17**	QDR SRAM Data In 17	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 17
C17	**D18**	QDR SRAM Data In 18	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 18
B15	**D19**	QDR SRAM Data In 19	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 19
D20	**D20**	QDR SRAM Data In 20	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 20
D17	**D21**	QDR SRAM Data In 21	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 21

A15	**D22**	QDR SRAM Data In 22	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 22
B21	**D23**	QDR SRAM Data In 23	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 23
B18	**D24**	QDR SRAM Data In 24	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 24
D16	**D25**	QDR SRAM Data In 25	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 25
B20	**D26**	QDR SRAM Data In 26	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 26
A19	**D27**	QDR SRAM Data In 27	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 27
E16	**D28**	QDR SRAM Data In 28	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 28
D19	**D29**	QDR SRAM Data In 29	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 29
D18	**D30**	QDR SRAM Data In 30	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 30
E17	**D31**	QDR SRAM Data In 31	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 31
C19	**D32**	QDR SRAM Data In 32	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 32
B19	**D33**	QDR SRAM Data In 33	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 33
E18	**D34**	QDR SRAM Data In 34	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 34
A20	**D35**	QDR SRAM Data In 35	(VDDQ, GND) / CMOS Input	The QDR Input Data Bus 35
A1	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
A8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
A9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
A10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
A22	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
B2	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
B8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
B9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.

C1	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
C3	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
C18	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
D1	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
D3	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
D10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
D12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
E7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
E9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
E11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
E13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F19	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
F21	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.

G13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
H20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
J17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
K16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.

K20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
L17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
M20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
N17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.

P10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
P20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
R17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T6	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T8	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T10	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T12	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T14	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
T20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U5	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U7	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U9	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U11	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.

U13	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U15	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
U17	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
V20	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
W1	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
Y19	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
AB1	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
AB2	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
AB16	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
AB22	**GND**	Digital Ground (CMOS)		Digital GND. All pins must be tied to single potential ground plane.
G1	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
G3	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
H2	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
M2	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
N1	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
N3	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
Y7	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
Y13	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
AA8	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
AA12	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
AB7	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.
AB13	**GNDA**	Analog Ground (CMOS)		Analog GND. All pins must be tied to single potential ground plane.

C7	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
C9	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
D5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
E2	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
E4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
F5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
H4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
H5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
K2	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
K4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
K5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
M4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
M5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
P5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
R2	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
R4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.

T1	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
T3	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
T5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V2	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V6	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V8	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V10	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V12	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V14	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
V16	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
W2	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
W5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
W8	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
W10	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
W12	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.

W15	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
Y4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
Y16	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
AA5	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
AA10	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
AA15	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
AB4	**GNDS**	SERDES Ground (CMOS)		Analog GND for TX/RX pairs. All pins must be tied to single potential ground plane.
A6	**ID0**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 0. This should be set statically to Vdd or GND at power-up. NOTE: SUPPLY / LEVELS REQUIREMENTS ARE UNQUE FROM THE OTHER I^2C PINS.
B6	**ID1**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 1. This should be set statically to Vdd or GND at power-up.
A7	**ID2**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 2. This should be set statically to Vdd or GND at power-up.
B7	**ID3**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 3. This should be set statically to Vdd or GND at power-up.
D6	**ID4**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 4. This should be set statically to Vdd or GND at power-up.
C8	**ID5**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 5. This should be set statically to Vdd or GND at power-up.
D8	**ID6**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 6. This should be set statically to Vdd or GND at power-up.
C10	**ID7**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 8. This should be set statically to Vdd or GND at power-up.
C11	**ID8**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 7. This should be set statically to Vdd or GND at power-up.
C12	**ID9**	I^2C	(VDD, GND) / CMOS Input	I^2C Slave ID address bit 9. This should be set statically to Vdd or GND at power-up.
U2	**IDS**	ID Select	(VDD, GND) / CMOS Input	sRIO 8/16 bit Destination ID Select
AA1	**IRQ0**	Interrupt 0	(VDD3, GND) / CMOS Output	This is an interrupt output pin whose value is given by the Error Management Block.

Y1	**IRQ1**	Interrupt 1	(VDD3, GND) / CMOS Output	This is an interrupt output pin whose value is given by the Error Management Block.
A11	**MBDONE**	Memory BIST	(VDD, GND) / CMOS Output	MBIST Done. Set (MBDONE = 1) when MBIST patterns are completed
B11	**MBPASS**	Memory BIST	(VDD, GND) / CMOS Output	MBIST Pass. Set (MBPASS = 1) when MBIST patterns pass. Cleared (MBPASS = 0) and is sticky when MBIST fails.
B3	**MRST_N**	Master Reset	(VDD, GND) / CMOS Input	SerB Global Reset. Sets all internal registers to default values. Resets all PLLs. Resets all port configurations. This is a HARD Reset.
Y17	**PLL_OFF**	PLL Off	(VDD, GND) / CMOS Input	Used for device testing with PLL bypass.
B10	**PPE_N**	Parallel Port Enable	(VDD, GND) / CMOS Input	PPE = 0, P-Port is active PPE = 1, P-Port is powered down and not used (low power).
P18	**Q0**	QDR SRAM Data Out 0	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 0
R18	**Q1**	QDR SRAM Data Out 1	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 1
P19	**Q2**	QDR SRAM Data Out 2	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 2
R19	**Q3**	QDR SRAM Data Out 3	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 3
T21	**Q4**	QDR SRAM Data Out 4	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 4
U21	**Q5**	QDR SRAM Data Out 5	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 5
T22	**Q6**	QDR SRAM Data Out 6	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 6
U22	**Q7**	QDR SRAM Data Out 7	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 7
N19	**Q8**	QDR SRAM Data Out 8	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 8
N18	**Q9**	QDR SRAM Data Out 9	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 9
R22	**Q10**	QDR SRAM Data Out 10	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 10
R21	**Q11**	QDR SRAM Data Out 11	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 11
P22	**Q12**	QDR SRAM Data Out 12	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 12
P21	**Q13**	QDR SRAM Data Out 13	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 13
N22	**Q14**	QDR SRAM Data Out 14	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 14
N21	**Q15**	QDR SRAM Data Out 15	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 15

M22	**Q16**	QDR SRAM Data Out 16	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 16
M21	**Q17**	QDR SRAM Data Out 17	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 17
L21	**Q18**	QDR SRAM Data Out 18	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 18
L22	**Q19**	QDR SRAM Data Out 19	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 19
K21	**Q20**	QDR SRAM- Data Out 20	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 20
K22	**Q21**	QDR SRAM Data Out 21	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 21
J21	**Q22**	QDR SRAM Data Out 22	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 22
J22	**Q23**	QDR SRAM Data Out 23	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 23
L18	**Q24**	QDRSRAM Data Out 24	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 24
L19	**Q25**	QDR SRAM Data Out 25	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 25
H21	**Q26**	QDR SRAM Data Out 26	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 26
H22	**Q27**	QDR SRAM Data Out 27	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 27
G21	**Q28**	QDR SRAM Data Out 28	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 28
G22	**Q29**	QDR SRAM Data Out 29	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 29
K18	**Q30**	QDR SRAM Data Out 30	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 30
K19	**Q31**	QDR SRAM Data Out 31	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 31
H19	**Q32**	QDR SRAM Data Out 32	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 32
J19	**Q33**	QDR SRAM Data Out 33	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 33
J18	**Q34**	QDR SRAM Data Out 34	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 34
G19	**Q35**	QDR SRAM Data Out 35	(VDDQ, GND) / CMOS Output	The QDR Output Data Bus 35
AA17	**QDRA**	QDR Mem Size	(VDD, GND) / CMOS Input	Reserved and should be tied to Ground.
AB18	**QDRB**	QDR Mem Size	(VDD, GND) / CMOS Input	Specifies what size QDR SRAM is connected externally. 0 = 16 address lines are active (36M QDR2 B4 SRAM) 1 = 17 address lines are active (72M QDR2 B4 SRAM)

V21	**RDO_N**	Read Strobe	(VDD, GND) / CMOS Output	When QDR type SRAM attached, this output should be connected to the /Rd input on the QDR SRAM(s). The FIFO controller will use this pin to control the read function on the SRAM.
AB3	**REFCLKN**	SERDES Clock	(VDD, GND) / Differential Input	Negative side of differential input clock. This clock is used as the 156MHz reference for standard SERDES operation.
AA3	**REFCLKP**	SERDES Clock	(VDD, GND) / Differential Input	Positive side of differential input clock. This clock is used as the 156MHz reference for standard SERDES operation.
V1	**REXTN**	Rext		External bias resistor. This pin must be connected to Rextp with a 12k Ohm resistor. This establishes the drive bias on the SERDES output. This provides CML driver stability across process and temperature.
U1	**REXTP**	Rext		External bias resistor. This pin must be connected to Rextn with a 12k Ohm resistor.
L1	**S1_RXN0**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Negative end of differential receiver, S-Port, Lane 0
M1	**S1_RXP0**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Positive end of differential receiver, S-Port, Lane 0
L4	**S1_RXN1**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Negative end of differential receiver, S-Port, Lane 1
L3	**S1_RXP1**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Positive end of differential receiver, S-Port, Lane 1
J1	**S1_RXN2**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Negative end of differential receiver, S-Port, Lane 2
H1	**S1_RXP2**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Positive end of differential receiver, S-Port, Lane 2
J4	**S1_RXN3**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Negative end of differential receiver, S-Port, Lane 3
J3	**S1_RXP3**	Port 1 Receive	(VDDS, GNDS) / S-Port 1 Differential Input	Positive end of differential receiver, S-Port, Lane 3
P1	**S1_TXN0**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Negative end of differential transmitter, S-Port, Lane 0
P2	**S1_TXP0**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Positive end of differential transmitter, S-Port, Lane 0
P4	**S1_TXN1**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Negative end of differential transmitter, S-Port, Lane 1
N4	**S1_TXP1**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Positive end of differential transmitter, S-Port, Lane 1
F1	**S1_TXN2**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Negative end of differential transmitter, S-Port, Lane 2
F2	**S1_TXP2**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Positive end of differential transmitter, S-Port, Lane 2
F4	**S1_TXN3**	Port1 Transmit	(VDDS, GNDS) / S-Port 1 Differential Output	Negative end of differential transmitter, S-Port, Lane 3
G4	**S1_TXP3**	Port1 Transmit	(VDDS, GNDS) / RIO Differential Output	Positive end of differential transmitter, S-Port, Lane 3

AB6	**S2_RXN0**	Port 2 Receive	(VDDS, GNDS)	Negative end of differential receiver, S-Port 2, Lane 0
AA6	**S2_RXP0**	Port 2 Receive	(VDDS, GNDS)	Positive end of differential receiver, S-Port 2, Lane 0
W6	**S2_RXN1**	Port 2 Receive	(VDDS, GNDS)	Negative end of differential receiver, S-Port 2, Lane 1
W7	**S2_RXP1**	Port 2 Receive	(VDDS, GNDS)	Positive end of differential receiver, S-Port 2, Lane 1
AB14	**S2_RXN2**	Port 2 Receive	(VDDS, GNDS)	Negative end of differential receiver, S-Port 2, Lane 2
AA14	**S2_RXP2**	Port 2 Receive	(VDDS, GNDS)	Positive end of differential receiver, S-Port 2, Lane 2
W14	**S2_RXN3**	Port 2 Receive	(VDDS, GNDS)	Negative end of differential receiver, S-Port 2, Lane 3
W13	**S2_RXP3**	Port 2 Receive	(VDDS, GNDS)	Positive end of differential receiver, S-Port 2, Lane 3
AB9	**S2_TXN0**	Port 2 Transmit	(VDDS, GNDS)	Negative end of differential transmitter, S-Port 2, Lane 0
AB8	**S2_TXP0**	Port 2 Transmit	(VDDS, GNDS)	Positive end of differential transmitter, S-Port 2, Lane 0
W9	**S2_TXN1**	Port 2 Transmit	(VDDS, GNDS)	Negative end of differential transmitter, S-Port 2, Lane 1
Y9	**S2_TXP1**	Port 2 Transmit	(VDDS, GNDS)	Positive end of differential transmitter, S-Port 2, Lane 1
AB11	**S2_TXN2**	Port 2 Transmit	(VDDS, GNDS)	Negative end of differential transmitter, S-Port 2, Lane 2
AB12	**S2_TXP2**	Port 2 Transmit	(VDDS, GNDS)	Positive end of differential transmitter, S-Port 2, Lane 2
W11	**S2_TXN3**	Port 2 Transmit	(VDDS, GNDS)	Negative end of differential transmitter, S-Port 2, Lane 3
Y11	**S2_TXP3**	Port 2 Transmit	(VDDS, GNDS)	Positive end of differential transmitter, S-Port 2, Lane 3
B12	**SCEN**	SCAN	(VDD, GND) / CMOS Input	SCAN Enable. SCAN is enabled when SCEN = 1. Scan clock is provided by SCK while SCEN = 1. Internal pull-down ensures disable if this pin is not driven.
C4	**SCL**	I^2C	(VDD3, GND) / CMOS Input	I^2C Clock. This is also repurposed for the SCAN clock when SCEN = 1.
C5	**SDA**	I^2C	(VDD3, GND) / CMOS IO	I^2C Serial Data IO. Data direction is determined by the I^2C Read/Write bit. See I^2C functionality for further detail.
W3	**SP1S0**	S-Port 1 Speed Select	(VDD, GND) / CMOS Input	Speed Select Pins. These pins define S-Port port speed at RESET for all ports. The RESET setting may be overridden by subsequent programming of the QUAD_CTRL register. SP1S[1:0] = {00 = 1.25G, 01 = 2.5G, 10 = 3.125G, 11 = RESERVED}. These pins must remain STATICALLY BIASED after power-up.
Y3	**SP1S1**	S-Port 1 Speed Select	(VDD, GND) / CMOS Input	Speed Select Pins. These pins define S-Port port speed at RESET for all ports.
AA2	**STOA**	SERDES Analog		SERDES Analog Test Output. Used for observing SERDES outputs.
Y2	**STOD**	SERDES Digital		SERDES Digital Test Output. Used for observing SERDES outputs.
B4	**TCK**	JTAG	(VDD3, GND) / CMOS Input	JTAG Tap Port Clock
B5	**TDI**	JTAG	(VDD3, GND) / CMOS Input	JTAG Tap Port Input
A5	**TDO**	JTAG	(VDD3, GND) / CMOS Output	JTAG Tap Port Output

A14	**TM0**	TMODE0	(VDD3, GND) / CMOS Input	TM[2:0]; MBIST Enable for use in testing on-chip memories. MBIST is enabled when mben = 1. Internal pull-down ensures disable if this pin is not driven.
B13	**TM1**	TMODE1	(VDD3, GND) / CMOS Input	TM[2:0]; MBIST Enable for use in testing on-chip memories.
A12	**TM2**	TMODE2	(VDD3, GND) / CMOS Input	TM[2:0]; MBIST Enable for use in testing on-chip memories.
A3	**TMS**	JTAG	(VDD3, GND) / CMOS Input	JTAG Tap Port Mode Select
A4	**TRST**	JTAG	(VDD3, GND) / CMOS Input	JTAG Tap Port Asynchronous Reset
C22	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
D2	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
D4	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
D11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
D13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
E20	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F17	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.

G10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
M7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.

M9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
M11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
M13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T7	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T9	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T11	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T13	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U4	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.

U6	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U8	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U10	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U12	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U14	**VDD**	1.2V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
A2	**VDD3**	2.5V / 3.3V JTAG Power (CMOS)		Digital JTAG Pin VDD3. All pins must be tied to single potential power supply plane.
B1	**VDD3**	2.5V / 3.3V JTAG Power (CMOS)		Digital JTAG Pin VDD3. All pins must be tied to single potential power supply plane.
C2	**VDD3**	2.5V / 3.3V JTAG Power (CMOS)		Digital JTAG Pin VDD3. All pins must be tied to single potential power supply plane.
F3	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
G2	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
H3	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
M3	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
N2	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
P3	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
Y6	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
Y8	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
Y12	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
Y14	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
AA7	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
AA13	**VDDA**	Analog Power (CMOS)		Analog VDD. All pins must be tied to single potential power supply plane.
A21	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.

C16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
C20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F18	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
F22	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
G20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
H17	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
J20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
K17	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
L20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
M15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
M17	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
N20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
P17	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
R16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.

R20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T15	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
T17	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U16	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
U20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
W20	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
AB21	**VDDQ**	1.5V Digital Power (CMOS)		Digital VDD. All pins must be tied to single potential power supply plane.
D7	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
D9	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
E1	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
E3	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
E5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
G5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
J2	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
J5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
K1	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
K3	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
L2	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
L5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
N5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
R1	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
R3	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.

R5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
T2	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
T4	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V7	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V9	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V11	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V13	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
V15	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
W4	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
W16	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
Y5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
Y10	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
Y15	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AA4	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AA9	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AA11	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AA16	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AB5	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AB10	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
AB15	**VDDS**	SERDES Power (CMOS)		Analog VDD for TX/RX pairs. All pins must be tied to single potential power supply plane.
A13	**VREF**	Reference Voltage (CMOS)		Toggle point reference voltage for HSTL inputs

U19	**WRO_N**	Write Strobe	(VDD, GND) / CMOS Output	When QDR type SRAM attached, this output should be connected to the /Wr input on the QDR SRAM(s). The FIFO controller will use this pin to control the write function on the SRAM.
AB17	**ZQ**	P-Port Impedance		
M18	**DNC**			Do Not Connect
M19	**DNC**			Do Not Connect
T18	**DNC**			Do Not Connect
T19	**DNC**			Do Not Connect

17.0 Package Specifications

17.1 Package Physical & Thermal Specifications

Package: Super FlipChip FCBGA(BR484)
Dimensions: 23 x 23mm
Ball Count: 484
Ball Diameter: 0.6mm
Ball Pitch: 1.0mm
Theta J_A = {11.9C/W @ 0m/s, 8 C/W @ 1m/s, 7.3 C/W @ 2m/s}
Theta J_C = 0.2 C/W

17.2 Package Drawing

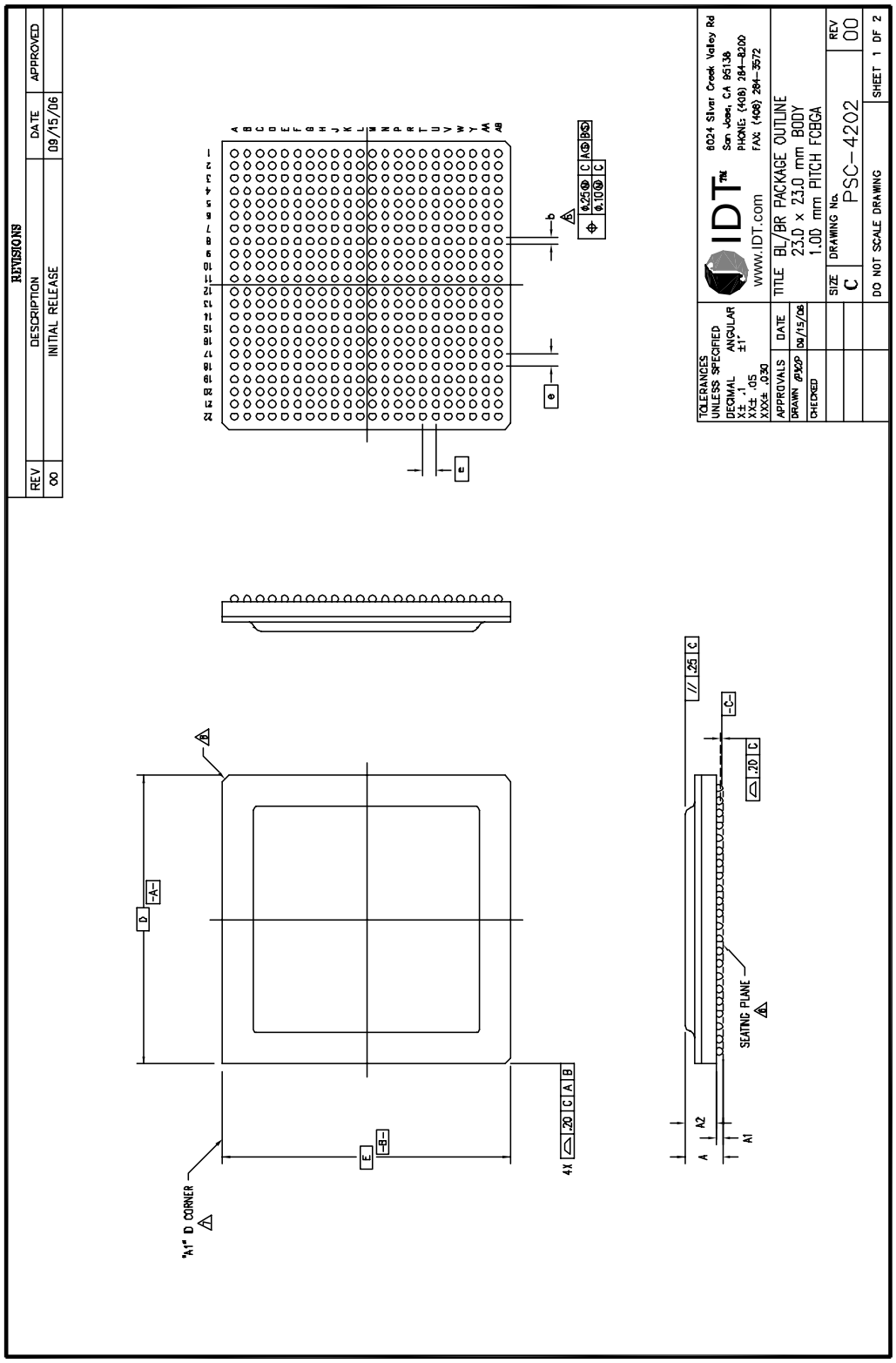


Figure 54 SerB Package Drawing 1 of 2

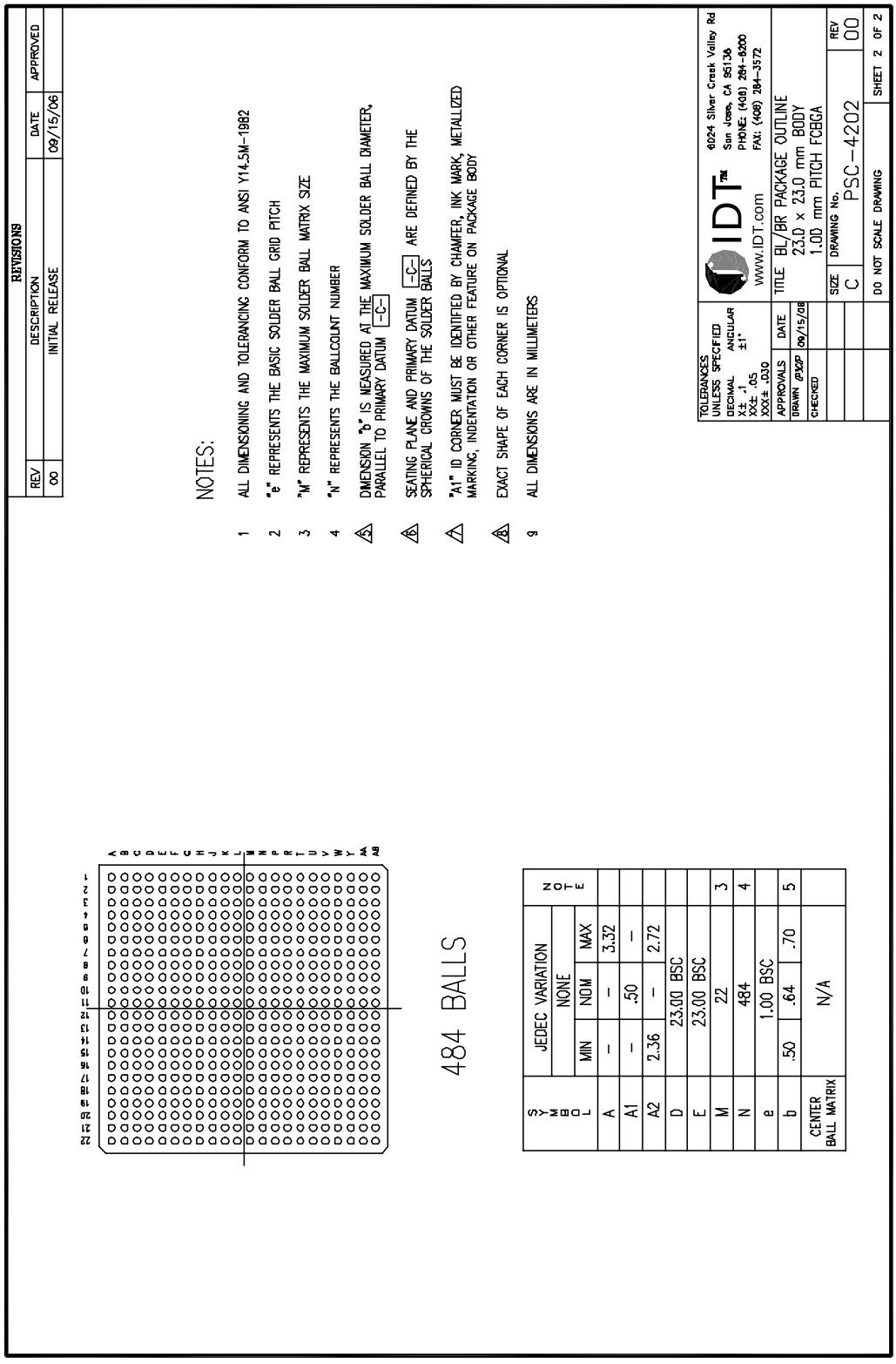


Figure 55 SerB Package Drawing 2 of 2

18.0 References and Standards

[1] "The I2C-Bus Specification", version 2.1, January 2000, Phillips

[2] RapidIO™ Interconnect Specification, Part 1: Input/Output Logical Specification, Rev. 1.3, 06/2005, RTA

[3] RapidIO™ Interconnect Specification, Part 2: Message Passing Logical Specification, Rev. 1.3, 06/2005, RTA

[4] RapidIO™ Interconnect Specification, Part 3: Common Transport Specification, Rev. 1.3, 06/2005, RTA

[5] RapidIO™ Interconnect Specification, Part 6: 1x/4x LP-Serial Physical Layer Specification, Rev. 1.3, 06/2005, RTA

[6] RapidIO™ Interconnect Specification, Part 7: System and Device Inter-operability Specification, Rev. 1.3, 06/2005, RTA

[7] RapidIO™ Interconnect Specification, Part 8: Error Management Extensions Specification, Rev. 1.3, 06/2005, RTA

[8] RapidIO™ Interconnect Specification, Part 9: Flow Control Logic Layer Extensions Specification, Rev. 1.3, 06/2005, RTA

[9] RapidIO™ Interconnect Specification, Annex I: Software/System Bring Up Specification, Rev. 1.3, 06/2005, RTA

[10] RapidIO™ Specification Revision 1.2: Errata 1, Rev. 1, 06/2003, RapidIO™ Trade Association

[11] IEEE Std 1149.1-2001 IEEE Standard Test Access Port and Boundary-Scan Architecture

[12] IEEE Std 1149.6-2003 IEEE Standard for Boundary-Scan Testing of Advanced Digital Networks

[13] QDR2 SRAM, Datasheet, Samsung, K7R163684B

[14] JEDEC Standard, JESD8-6 HSTL

19.0 Revision History

10/06/06: Initial Advanced Datasheet (Rev A)

03/01/07: Advanced Datasheet (Rev B)

19.1 Advanced Datasheet: (Definition)

"ADVANCED datasheet contain descriptions for products that are in early release. "Advanced datasheets are informational only. Advanced specifications are subject to change without notice.

20.0 Ordering Information

For specific speeds, packages and powers, contact your sales office

CORPORATE HEADQUARTERS
6024 Silver Creek Valley Road
San Jose, CA 95138

for SALES:
800-345-7015 or 408-284-8200
fax: 408-284-2775
www.idt.com

for Tech Support:
831-284-2794
DualPortHelp@idt.com