Input/Output

HMM-145-0

Peer-to-peer Messages

Peer-to-peer messaging is a GigaRing channel communication protocol. When a GigaRing channel client uses peer-to-peer messaging, it creates a GigaRing message packet and sends the packet to the destination GigaRing channel client. The destination GigaRing client then processes the message payload portion of the GigaRing message packet.

Sending a GigaRing Message Packet

Any PE in the system can send a GigaRing message to a destination GigaRing channel client. The following steps describe this process.

1. The microprocessor in a PE stores a GigaRing message packet in local memory.

   There are two forms of GigaRing message packets: response and request (refer to Table 1 and Table 2). The response packet is used for peer-to-peer messages in present-day Cray Research products. The request packet will be used for future products that support request and response messaging protocol.

Table 1. GigaRing Message Response Packet Format

<table>
<tr><td></td><td colspan="6">Bits <31 : 24>   Bits <23 : 16>   Bits <15 : 8>   Bits <7 : 0></td></tr>
<tr><td rowspan="4">Header</td><td>Target ID <12 : 0></td><td>D</td><td>E</td><td>1 0 0 1 0 0</td><td>Length/Err <5 : 0></td><td>Flow <4 : 0></td></tr>
<tr><td>Source ID <12 : 0></td><td>C</td><td colspan="3">Sequence <12 : 0></td><td>Control <4 : 0></td></tr>
<tr><td colspan="6">Software-defined Message Information <31 : 0> (Not used by hardware)</td></tr>
<tr><td colspan="6">Software-defined Message Information <31 : 0> (Not used by hardware)</td></tr>
<tr><td>Data</td><td colspan="6">0 or more 64-bit words of Message Payload
(32 words maximum)</td></tr>
</table>

NOTE: Software may store a different type of GigaRing channel packet in memory instead of a GigaRing message packet. For example, software may store a GigaRing maintenance packet or a GigaRing memory-mapped register read packet.

Table 2. GigaRing Message Request Packet Format

	Bits <31 : 24>	Bits <23 : 16>			Bits <15 : 8>	Bits <7 : 0>
Header	Target ID <12 : 0>	D	E	0 1 0 1 0 0	Length/Err <5 : 0>	Flow <4 : 0>
Header	Source ID <12 : 0>	C	Sequence <12 : 0>			Control <4 : 0>
Header	Software-defined Message Information <31 : 0> (Not used by hardware)					
Header	Software-defined Message Information <31 : 0> (Not used by hardware)					
Data	0 or more 64-bit words of Message Payload (32 words maximum)					

2. The microprocessor stores message-send request parameters into a block of eight contiguous E registers. These E registers will serve as the source E registers for a SEND command. Table 3 shows the bit format of the block of eight source E registers for a message-send request.

Table 3. Message-send Request Source E-registers Bit Format

E-register	Bits <63 : 56>	Bits <55 : 48>	Bits <47 : 40>	Bits <39 : 32>	Bits <31 : 24>	Bits <23 : 16>	Bits <15 : 8>	Bits <7 : 0>
n	Logical PE Number		Global Virtual Address that Points to the GigaRing Message Packet in Memory					
n + 1	Must be Zero (MBZ)						Ex	Length
n + 2	MBZ							
n + 3	MBZ							
n + 4	MBZ							
n + 5	MBZ							
n + 6	MBZ							
n + 7	MBZ							

The logical PE number and global virtual address in the first source E register indicate the location of the GigaRing message packet in system memory. The logical PE number indicates the logical PE in which software stored the message packet. The global virtual address indicates the location of the message packet in the local memory of that PE.

The length field in the second source E register indicates the number of 64-bit words in the message packet. For example, when software creates a message packet with a 10-word payload, software sets the length field to 12. This value includes the 10-word payload and the 2-word header of the message packet (refer again to Table 1). If software does not set the length correctly, undefined results may occur.

The Ex field in the second source E register indicates whether the message packet is for an express GigaRing memory-mapped register operation.

3. The microprocessor issues a SEND E-register command to the outgoing message queue (I_MSG_OUT_Q) register of one of the I/O controllers in the system.

   Refer again to Table 3 for the bit format of the source E registers for the SEND command.

   The outgoing message queue can store one active GigaRing message send request plus up to eight outgoing GigaRing message-send requests. The I/O controller processes the GigaRing message-send requests in the order it receives E-register SEND commands.

   If the outgoing message queue is full, the I/O controller sends a no-acknowledge for the E-register SEND command back to the requesting PE. This changes the E-register state (ERS) code for the source E registers to the full and send rejected (FULL_SR) state. The requesting PE then resends the message-send request to an I/O controller.

4. The I/O controller sends the GigaRing message packet to the destination client on the GigaRing channel. The I/O controller also sends an acknowledge for the E-register SEND command back to the requesting PE.

Receiving a GigaRing Message Packet

Any PE in the system may receive a GigaRing message packet from a GigaRing client. The following steps describe this process.

1. The I/O controller receives a GigaRing message packet from a GigaRing client. Refer again to Table 1 for the bit format of a GigaRing message packet.

2. The I/O controller determines how many 64-bit words of data are in the message payload of the GigaRing message packet.

   If the payload is 0 words or if the message packet is an express GigaRing register read packet, the message does not contain a payload. In this case, the I/O controller continues to process the message header information (refer to step 6).

3. If the message payload is 1 or more 64-bit words, the I/O controller determines whether there is room in a reserved location of system memory for storing GigaRing message payloads.

   To do this, the I/O controller reads the limit value (bits <63 : 52>) of the incoming message address (I_MSG_IN_AD) register. If the limit value is greater than 0, there is room in system memory for the payload. If the limit value is 0, the I/O controller determines what process to perform by reading the value of the drop bit (bit 7) of the incoming message control (I_MSG_IN_CTL) register.

   If the drop bit is set to 0, the I/O controller holds the payload and stalls incoming messages until software resets the limit value in the I_MSG_IN_AD register. If the drop bit is set to 1, the I/O controller destroys the payload of the message, ignores the header, and continues with messaging operations.

4. If there is room in the reserved location of system memory for storing GigaRing message payloads, the I/O controller determines the location for the payload in system memory.

   To do this, the I/O controller reads the logical PE number and global virtual address (GVA) values stored in the I_MSG_IN_AD register. The logical PE number identifies the PE that stores message payloads, and the global virtual address identifies a location in the local memory of that PE.

5. The I/O controller writes the payload to the system memory locations identified in step 4. To do this, the I/O controller performs PUT and/or PUTV E-register commands to the destination memory locations.

6. The I/O controller transfers the header of the GigaRing message packet to a PE that software identifies as the controlling PE.

   First, the I/O controller determines which logical PE is the controlling PE. To do this, the I/O controller reads address values stored in the incoming message-queue control word address (I_MSG_IN_MQCW_AD) register.

   The address values contain a logical PE number and a global virtual address. The logical PE number identifies the controlling PE. The global virtual address identifies the location of the destination MQCW in the local memory of the controlling PE.

   Second, the I/O controller transfers the GigaRing message-packet header and additional control information to the controlling PE. To do this, the I/O controller issues a SEND E-register command to the controlling PE. The controlling PE stores the header and additional control information in its message queue.

Table 4 shows the bit format of the GigaRing message-packet header information in a slot of the controlling PE message queue.

Table 4. GigaRing Message-packet Header in a Message Queue Slot

<table>
<tr><th>Message Queue Slot Bits</th><th>Bits <63 : 56></th><th colspan="2">Bits <55 : 48></th><th colspan="2">Bits <47 : 40></th><th colspan="2">Bits <39 : 32></th><th>Bits <31 : 24></th><th colspan="3">Bits <23 : 16></th><th colspan="2">Bits <15 : 8></th><th colspan="2">Bits <7 : 0></th></tr>
<tr><td><63 : 00></td><td colspan="12">I_MSG_IN_AD Register Bits <63 : 9></td><td>Ex</td><td colspan="2">Length</td></tr>
<tr><td><127 : 64></td><td colspan="2">Target ID</td><td colspan="2">D E 100100</td><td colspan="2">Length</td><td>Flow</td><td colspan="2">Source ID</td><td>C</td><td colspan="4">Sequence</td><td>Control</td></tr>
<tr><td><191 : 128></td><td colspan="7">Message Information <63 : 32></td><td colspan="8">Message Information <31 : 0></td></tr>
<tr><td><255 : 192></td><td colspan="15">Bits <63 : 0> are not used (RAZ) unless Ex bit = 1
When Ex = 1, these bits contain GigaRing register read response data.</td></tr>
<tr><td><319 : 256></td><td colspan="15">Bits <63 : 0> are not used (RAZ)</td></tr>
<tr><td><383 : 320></td><td colspan="15">Bits <63 : 0> are not used (RAZ)</td></tr>
<tr><td><447 : 384></td><td colspan="15">Bits <63 : 0> are not used (RAZ)</td></tr>
<tr><td><511 : 448></td><td colspan="15">Bits <63 : 0> are not used (RAZ)</td></tr>
</table>

NOTE: Table 4 shows the header format of a GigaRing message-response packet in a message queue slot. The GigaRing packet header can be from a different type of GigaRing packet. For example, the header could be from a maintenance-read GigaRing packet.

7. If the message payload was 1 or more 64-bit words, the I/O controller increments the global virtual address bits (bits <37 : 9>) of the I_MSG_IN_AD register by one and decrements the limit bits (bits <63 : 52>) of the I_MSG_IN_AD register by one.

   NOTE: The I/O controller performs these steps in an indivisible operation; therefore, software cannot accidentally modify the value of the I_MSG_IN_AD register while the I/O controller performs a GigaRing message-packet receive operation.

8. After receiving the GigaRing message-packet header and additional control information from the I/O controller, the microprocessor in the controlling PE reads the message length, I_MSG_IN_AD register information (logical PE number and global virtual address), and GigaRing packet header information out of its message queue. The microprocessor uses this information to process the GigaRing message-packet header and locate the GigaRing message-packet payload.

Master DMA Transfers

A device on a GigaRing channel ring may read from or write to the memory in another device on the GigaRing channel ring using direct memory access (DMA) transfers. In this document, when a PE in the CRAY T3E system initiates the DMA transfer, it is called a “master” DMA transfer.

Performing a Master DMA Read

Any PE in the system can read directly from the memory of a destination GigaRing channel client. The following steps describe this process.

1. The microprocessor in a PE stores a GigaRing read-block-initialization packet in local memory.

   The read-block-initialization packet provides base address and length information for the read transfer (refer to Table 5).

Table 5. GigaRing Read-block-initialization Packet

	Bits <31 : 24>	Bits <23 : 16>			Bits <15 : 8>	Bits <7 : 0>
Header	Target ID <12 : 0>	D	E	0 0 0 1 0 0	Length/Err <5 : 0>	Flow <4 : 0>
Header	Source ID <12 : 0>	C	Sequence <12 : 0>			Control <4 : 0>
Header	Slave Base Address <63 : 32>					
Header	Slave Base Address <31 : 0>					
Data	Length <31 : 0>					
Data	These bits are not used <31 : 0>					
Data	Slave-specific Auxiliary Data Block					

2. The microprocessor sends a read-block transfer initiation message (TIM) to an I/O controller.

   To do this, the microprocessor issues a SEND E-register command to the master DMA read request queue (I_MDMA_RD_Q) register of the I/O controller that will perform the DMA read transfer. Table 6 shows the bit format of the block of eight source E-registers for a SEND E-register command to an I_MDMA_RD_Q register.

Table 6. Read Block TIM Source E-register Bit Formats

E-register	Bits <63 : 56>	Bits <55 : 48>	Bits <47 : 40>	Bits <39 : 32>	Bits <31 : 24>	Bits <23 : 16>	Bits <15 : 8>	Bits <7 : 0>
n	Logical PE Number		Global Virtual Address that points to GigaRing Read-block-initialization Packet					
n + 1	Identification		0000	Control Logical PE	00	Control PE MQCW GVA Bits <37 : 8>		
n + 2	000000	DMA Read Length			000000	DMA Read Stride		
n + 3	Not used (MBZ)		DMA Read Global Index <49 : 2>					
n + 4	Not used (MBZ)		DMA Read Base <49 : 2>					
n + 5	Not used (MBZ)		DMA Read Mask <49 : 2>					
n + 6	Not used (MBZ)							
n + 7	Not used (MBZ)							

The logical PE number and global virtual address in the first source E register provide the location of the GigaRing read-block-initialization packet. The logical PE number indicates the logical PE in which software stored the packet. The global virtual address indicates the location of the packet in the local memory of that PE.

The control logical PE and control PE MQCW GVA values in the second source E register provide the location of the message queue in the controlling PE. The logical PE number indicates which logical PE is the controlling PE. The MQCW GVA identifies the location of the MQCW in the local memory of the controlling PE. The I/O controller needs this information so that when the master DMA read operation is complete, the I/O controller can send control information to the controlling PE.

The length, stride, global index, base, and mask values in the third, fourth, fifth, and sixth E registers are initial values for the master DMA read engine in the I/O controller. The master DMA read engine uses these values to calculate the addresses in CRAY T3E system memory where data from the GigaRing client will be stored.

3. The I/O controller acknowledges the read-block TIM and transfers the GigaRing read-block-initialization packet from CRAY T3E system memory to the GigaRing client.

4. The GigaRing client receives the read-block-initialization packet, creates a read-block-initialization-response packet, and sends the packet to the I/O controller that requested the transfer.

5. The I/O controller sends multiple GigaRing read packets to the GigaRing client.

   The I/O controller uses the slave address and control information from the read-block-initialization-response packet and the master address and control information generated by the DMA read engine to create GigaRing read packets. The GigaRing client sends GigaRing read-response-packets to the I/O controller for each read packet it receives.

6. At the end of the transfer, the I/O controller sends a GigaRing read-block-done packet to the GigaRing client.

7. The GigaRing client sends a GigaRing read-block-done-response packet to the I/O controller.

8. The I/O controller transfers the GigaRing read-block-done-response packet to a PE that software identified as the controlling PE.

   First, the I/O controller determines which logical PE is the controlling PE. To do this, the I/O controller uses the control PE address values it received from the read-block TIM (refer again to Table 6).

Second, the I/O controller transfers the GigaRing read-block-done-response packet to the controlling PE. To do this, the I/O controller issues a SEND E-register command to the MQCW of the controlling PE. Table 10 shows the bit format of the GigaRing read-block-done-response packet information in a slot of the controlling PE message queue.

Table 7. GigaRing Read-block-done-response Packet in Message Queue Slot

Message Queue Slot Bits	Bits <63 : 56>	Bits <55 : 48>	Bits <47 : 40>	Bits <39 : 32>	Bits <31 : 24>	Bits <23 : 16>	Bits <15 : 8>	Bits <7 : 0>
<63 : 00>	Bits <63 : 0> are not used (RAZ)							
<127 : 64>	Target ID	D E 100100	Length	Flow	Source ID	C	Sequence	Control
<191 : 128>	Bits <63 : 32> are an unknown value				Bits <31 : 0> are an unknown value			
<255 : 192>	Bits <63 : 0> are not used (RAZ)							
<319 : 256>	Bits <63 : 0> are not used (RAZ)							
<383 : 320>	Bits <63 : 0> are not used (RAZ)							
<447 : 384>	Bits <63 : 0> are not used (RAZ)							
<511 : 448>	Bits <63 : 0> are not used (RAZ)							

9. The controlling PE reads the GigaRing read-block-done-response packet out of the message queue. This action completes the DMA read block operation.

Performing a Master DMA Write

Any PE in the system can write directly to the memory of a destination GigaRing channel client. The following steps describe this process.

1. The microprocessor in a PE stores a GigaRing write-block-initialization packet in local memory.

   The write-block-initialization packet provides base address and length information for the write transfer (refer to Table 8).

Table 8. GigaRing Write-block-initialization Packet

<table>
<tr><th></th><th colspan="8">Bits <31 : 24></th><th colspan="8">Bits <23 : 16></th><th colspan="8">Bits <15 : 8></th><th colspan="8">Bits <7 : 0></th></tr>
<tr><td rowspan="4">Header</td><td colspan="13">Target ID <12 : 0></td><td>D</td><td>E</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td colspan="6">Length/Err <5 : 0></td><td colspan="5">Flow <4 : 0></td></tr>
<tr><td colspan="13">Source ID <12 : 0></td><td>C</td><td colspan="13">Sequence <12 : 0></td><td colspan="5">Control <4 : 0></td></tr>
<tr><td colspan="32">Slave Base Address <63 : 32></td></tr>
<tr><td colspan="32">Slave Base Address <31 : 0></td></tr>
<tr><td rowspan="3">Data</td><td colspan="32">Length <31 : 0></td></tr>
<tr><td colspan="32">These bits are not used <31 : 0></td></tr>
<tr><td colspan="32">Slave-specific Auxiliary Data Block</td></tr>
</table>

2. The microprocessor sends a write-block transfer initiation message (TIM) to an I/O controller.

   To do this, the microprocessor issues a SEND E-register command to the master DMA write request queue (I_MDMA_WR_Q) register of the I/O controller that will perform the DMA write transfer. Table 9 shows the bit format of the block of eight source E-registers for a SEND E-register command to an I_MDMA_WR_Q register.

Table 9. Write Block TIM Source E-register Bit Formats

<table>
<tr><th>E-register</th><th>Bits <63 : 56></th><th>Bits <55 : 48></th><th>Bits <47 : 40></th><th>Bits <39 : 32></th><th>Bits <31 : 24></th><th>Bits <23 : 16></th><th>Bits <15 : 8></th><th>Bits <7 : 0></th></tr>
<tr><td>n</td><td colspan="2">Logical PE Number</td><td colspan="6">Global Virtual Address that points to GigaRing Write-block-initialization Packet</td></tr>
<tr><td>n + 1</td><td colspan="2">Identification</td><td>0000</td><td>Control Logical PE</td><td>00</td><td colspan="3">Control PE MQCW GVA Bits <37 : 8></td></tr>
<tr><td>n + 2</td><td>000000</td><td colspan="3">DMA Write Length</td><td>000000</td><td colspan="3">DMA Write Stride</td></tr>
<tr><td>n + 3</td><td colspan="2">Not used (MBZ)</td><td colspan="6">DMA Write Global Index <49 : 2></td></tr>
<tr><td>n + 4</td><td colspan="2">Not used (MBZ)</td><td colspan="6">DMA Write Base <49 : 2></td></tr>
<tr><td>n + 5</td><td colspan="2">Not used (MBZ)</td><td colspan="6">DMA Write Mask <49 : 2></td></tr>
<tr><td>n + 6</td><td colspan="8">Not used (MBZ)</td></tr>
<tr><td>n + 7</td><td colspan="8">Not used (MBZ)</td></tr>
</table>

The logical PE number and global virtual address in the first source E register provide the location of the GigaRing write block initialization packet. The logical PE number indicates the logical PE in which software stored the packet. The global virtual address indicates the location of the packet in the local memory of that PE.

The control logical PE and control PE MQCW GVA values in the second source E register provide the location of the message queue in the controlling PE. The control logical PE number indicates which logical PE is the controlling PE. The MQCW GVA identifies the location of the MQCW in the local memory of the controlling PE. When the master DMA write operation is complete, the I/O controller sends control information to the controlling PE.

The length, stride, global index, base, and mask values in the third, fourth, fifth, and sixth E registers are initial values for the master DMA write engine in the I/O controller. The master DMA write engine uses these values to calculate the addresses in CRAY T3E system memory that store data for each GigaRing write packet.

3. The I/O controller acknowledges the write-block TIM and transfers the GigaRing write-block-initialization packet from CRAY T3E system memory to the GigaRing client.

4. The GigaRing client receives the write-block-initialization packet, creates a write-block-initialization-response packet, and sends the packet to the I/O controller that requested the transfer.

5. The I/O controller sends multiple GigaRing write packets to the GigaRing client.

   The I/O controller uses the slave address and control information from the write-block-initialization-response packet and the master address and control information generated by the DMA write engine to create GigaRing write packets. The GigaRing client sends GigaRing write-response-packets to the I/O controller for each write packet it receives.

6. At the end of the transfer, the I/O controller sends a GigaRing write-block-done packet to the GigaRing client.

7. The GigaRing client sends a GigaRing write-block-done-response packet to the I/O controller.

8. The I/O controller transfers the GigaRing write-block-done-response packet to a PE that software identified as the controlling PE.

   First, the I/O controller determines which logical PE is the controlling PE. To do this, the I/O controller uses the control PE address values it received from the write-block TIM (refer again to Table 9).

Second, the I/O controller transfers the GigaRing write-block-done-response packet to the controlling PE. To do this, the I/O controller issues a SEND E-register command to the MQCW of the controlling PE. Table 10 shows the bit format of the GigaRing write-block-done-response packet information in a slot of the controlling PE message queue.

Table 10. GigaRing Write-block-done-response Packet in Message Queue Slot

Message Queue Slot Bits	Bits <63 : 56>	Bits <55 : 48>	Bits <47 : 40>	Bits <39 : 32>	Bits <31 : 24>	Bits <23 : 16>	Bits <15 : 8>	Bits <7 : 0>
<63 : 00>	Bits <63 : 0> are not used (RAZ)							
<127 : 64>	Target ID	D E 100100	Length	Flow	Source ID	C	Sequence	Control
<191 : 128>	Current slave address bits <63 : 32>				Current slave address bits <31 : 0>			
<255 : 192>	Bits <63 : 0> are not used (RAZ)							
<319 : 256>	Bits <63 : 0> are not used (RAZ)							
<383 : 320>	Bits <63 : 0> are not used (RAZ)							
<447 : 384>	Bits <63 : 0> are not used (RAZ)							
<511 : 448>	Bits <63 : 0> are not used (RAZ)							

9. The controlling PE reads the GigaRing write-block-done-response packet out of the message queue. This action completes the DMA write block operation.

Slave DMA Transfers

A device on a GigaRing channel ring may read from or write to the memory in another device on the GigaRing channel ring using direct memory access (DMA) transfers. In this document, when the CRAY T3E system receives a DMA transfer request it is called a “slave” DMA transfer.

Performing a Slave DMA Read

The I/O controller performs all operations for a slave DMA read transfer. None of the PEs in the CRAY T3E system are involved with a slave DMA read transfer.

To initiate a slave DMA read transfer, a GigaRing client sends a GigaRing read-block-initialization packet to an I/O controller. Table 11 shows the bit format of a GigaRing read-block-initialization packet.

Table 11. GigaRing Read-block-initialization Packet

	Bits <31 : 24>	Bits <23 : 16>			Bits <15 : 8>		Bits <7 : 0>
Header	Target ID <12 : 0>		D	E	0 0 0 1 0 0	Length/Err <5 : 0>	Flow <4 : 0>
Header	Source ID <12 : 0>		C	Sequence <12 : 0>			Control <4 : 0>
Header	Global Index <63 : 32>						
Header	Global Index <31 : 0>						
Data	Length <31 : 0>						
Data	Stride <31 : 0>						
Data	Centrifuge Mask <63 : 32>						
Data	Centrifuge Mask <31 : 0>						
Data	Base Address <63 : 32>						
Data	Base Address <31 : 0>						

The I/O controller contains buffer sets that support 32 separate incoming slave DMA data streams. Each of these streams can be either a DMA read or a DMA write stream. When the I/O controller receives a DMA read-block-initialization packet, it checks to see if one of the 32 buffer sets is free. If none of the buffer sets are free, the I/O controller sends a GigaRing block response packet to the GigaRing client to indicate that the DMA read operation cannot be performed at this time.

If a buffer set is free, the I/O controller sends a GigaRing read-block-initialization response packet to the GigaRing client. Also, the I/O controller uses the length, stride, global index, base address, and mask values from the GigaRing read-block-initialization packet as initial parameters for the slave DMA read engine in the I/O controller. The slave DMA read engine uses these values to calculate the addresses in CRAY T3E system memory where data that the GigaRing client requested is stored. The slave DMA read engine uses GET and GETV E-register commands to retrieve the data from system memory.

The GigaRing client sends multiple GigaRing read packets to the I/O controller. The I/O controller uses the address and control information from the GigaRing read packets and data retrieved by the slave DMA read engine to create GigaRing read-response packets for all of the read packets it receives.

At the end of the transfer, the GigaRing client sends a GigaRing read-block-done packet to the I/O controller. The I/O controller then sends a GigaRing read-block-done-response packet to the GigaRing client.

Performing a Slave DMA Write

The I/O controller performs all operations for a slave DMA write transfer. None of the PEs in the CRAY T3E system are involved with a slave DMA write transfer.

To initiate a slave DMA write transfer, a GigaRing client sends a GigaRing write-block-initialization packet to an I/O controller. Table 12 shows the bit format of a GigaRing write-block-initialization packet.

Table 12. GigaRing Write-block-initialization Packet

<table>
<tr><th></th><th colspan="8">Bits <31 : 24></th><th colspan="8">Bits <23 : 16></th><th colspan="8">Bits <15 : 8></th><th colspan="8">Bits <7 : 0></th></tr>
<tr><td rowspan="4">Header</td><td colspan="13">Target ID <12 : 0></td><td>D</td><td>E</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td>0</td><td colspan="6">Length/Err <5 : 0></td><td colspan="5">Flow <4 : 0></td></tr>
<tr><td colspan="13">Source ID <12 : 0></td><td>C</td><td colspan="13">Sequence <12 : 0></td><td colspan="5">Control <4 : 0></td></tr>
<tr><td colspan="32">Global Index <63 : 32></td></tr>
<tr><td colspan="32">Global Index <31 : 0></td></tr>
<tr><td rowspan="6">Data</td><td colspan="32">Length <31 : 0></td></tr>
<tr><td colspan="32">Stride <31 : 0></td></tr>
<tr><td colspan="32">Centrifuge Mask <63 : 32></td></tr>
<tr><td colspan="32">Centrifuge Mask <31 : 0></td></tr>
<tr><td colspan="32">Base Address <63 : 32></td></tr>
<tr><td colspan="32">Base Address <31 : 0></td></tr>
</table>

The I/O controller contains buffer sets that support 32 separate incoming slave DMA data streams. Each of these streams can be either a DMA read or a DMA write stream. When the I/O controller receives a DMA write-block-initialization packet, it checks to see if one of the 32 buffer sets is free. If none of the buffer sets are free, the I/O controller sends a GigaRing block response packet to the GigaRing client to indicate that the DMA write operation cannot be performed at this time.

If a buffer set is free, the I/O controller sends a GigaRing write-block-initialization response packet to the GigaRing client. Also, the I/O controller uses the length, stride, global index, base address, and mask values from the GigaRing write-block-initialization packet as initial parameters for the slave DMA write engine in the I/O controller. The slave DMA write engine uses these values to calculate the addresses where data will be written in CRAY T3E system memory.

The GigaRing client sends multiple GigaRing write packets to the I/O controller. The I/O controller uses the data and control information from the GigaRing write packets and address information generated by the slave DMA write engine to write data into CRAY T3E system memory using PUT or PUTV E-register commands.

At the end of the transfer, the GigaRing client sends a GigaRing write-block-done packet to the I/O controller. The I/O controller then sends a GigaRing write-block-done-response packet to the GigaRing client.

Index