6
Remote memory access protocol (normative)

6.1 General

6.2 Definitions

Destination Address bytes are any leading address bytes required to reach the destination node on the SpaceWire network. These destination address bytes are stripped off by the time the packet reaches the destination. If logical addressing is being used there may be no destination address bytes present when the packet is transmitted.

Logical Address byte is the logical address of the destination. This may be used to route the packet to the destination or, if path addressing is being used, to simply confirm that the final destination is the correct one i.e. that the logical address of the destination matches the logical address in the packet. If the logical address of the destination is unknown then the default logical address of 254 may be used. The destination may chose to accept or reject packets with a logical address of 254.

Protocol Identifier byte identifies the particular protocol being used for communication. For the Remote Memory Access protocol the protocol identifier has the value 1 (01h) TBC.

Packet Type, Command, Return Address Length byte determines the type of the packet i.e. a command, a response or an acknowledgement. This byte also includes two bits that determine the number of extra 4-byte return addresses. For example, if these bits are set to the value two then there will be eight extra return address bytes. If they are set to zero then there are no extra address bytes.

Options this is a reserved field that may be used to provide options to the command byte.

Extra Return Address bytes provide extra bytes for the return address. This allows the return address to be longer than a single logical address byte and enables path addressing and regional logical addressing to be used in the return address. Leading zeros of the return address are ignored. If a packet is to be sent to address zero then this is done by setting all the extra return address bytes to zero. This will result in a single zero address byte being sent in front of the return address.

Return Address byte is the logical address to which the destination node for a command is to reply. The Return Address is normally set to the logical address of the source node that is sending the command. The Return Address byte may be set to 254 (0FEh) which is the default logical address, if the command source node does not have a logical address.

Transaction Identifier bytes are used to identify command, response, and acknowledge transactions that make up a particular read, write, indirect read or indirect write operation. The source of the command gives the command a unique transaction identify.

This transaction identifier is returned in the response or acknowledgement to the command. This allows the command source to send many commands without having to wait for a response to each command before sending the next command. When a response or acknowledge comes in it can be quickly associate with the command that caused it by the transaction identifier.

Write Address bytes form the 32-bit memory/register address to which the data in a write command is to be written. A write address is also used in an indirect read command to specify where in the memory of the intermediate node the data that is read from the target destination is to be written. Input/output registers and control/status registers are assumed to be memory mapped.

Data Length bytes form the 24-bit length of the data that is to be written or read. The length is the length in bytes.

Header Checksum byte is an 8-bit checksum used to confirm that the header is correct before executing the command.

Data bytes are the data that is to be written in a write command or the data that is read in a read response.

Data Checksum byte an 8-bit checksum used to confirm that the data was correctly transferred.

EOP character is the End Of Packet market of the SpaceWire packet.

6.3 Direct Write

6.3.1 Write command

The write command provides a means for one node, the source node, to write one or more bytes of data into memory of another node, the destination or target node on a SpaceWire network. The format of the command is shown in Figure 1.

Notes for people reviewing this document following previous discussions:

Moved logical Return address to end of extra return address since then can just stream them out in order. i.e. the return logical address is always at the end.

Data checksum rather than packet checksum because can do this checksum completely separately from header information.

First byte transmitted

	Destination Address	Destination Address	Destination Address
Logical Address	Protocol Identifier	Packet Type, Command Return Address Extra Length	Options
Extra Return Address	Extra Return Address	Extra Return Address	Extra Return Address
Return Address	Transaction Identifier	Transaction Identifier	Transaction Identifier
Write Address	Write Address	Write Address	Write Address
Data Length	Data Length	Data Length	Header Checksum
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data Checksum	EOP	

Last byte transmitted

Bits in Packet Type / Command / Return Address Extra Length Byte

Command/ Response	Read/ Write	Direct/ Indirect	Increment/ No inc. source	Increment/ No inc. target	Ack/ No ack	Extra Return Addr Words	Extra Return Addr Words

Figure 1 Write Command Format

The Destination Address is the address on the SpaceWire network of the node that is to have data written into its memory or registers. The destination address is made up of two parts: the Destination Address bytes which are optional (shaded in Figure 1) and the Logical Address. If path addressing is being used then the Destination Address bytes contain the path to the destination node. The Logical Address is byte is then set to the logical address of the destination node or to the default value 254 (0FEh). If logical addressing is being used there are no Destination Address bytes and the logical address is set to the logical address of the destination node. Normally logical addressing would be used and there would be no Destination Address bytes.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is set (1) indicate that the packet is a command packet, rather than a reply packet.

The Command field hold the direct write command.

The command option “Increment / No Increment Source” is not used by the write command and should be set to zero.

The command option “Increment / No Increment Target” is used for multiple data byte transfers. If set (1) it causes the write address in the target to increment on every byte (or word as determined by the target unit) so that data bytes are written to consecutive memory locations. If not set (0) the write address is not incremented so successive data bytes (or words as determined by the target unit) are written to the same memory location. Note that the width of the memory word is determined by the target unit and can be any multiple of 8-bits. For example, if the width of the target unit memory word is 32-bits then four data bytes from the data field of the command are written into one memory location in the target unit.

NOTE DO WE WANT TO ALLOW PADDING OF 8-BITS INTO 16-BIT ETC FIELD? IF SO WE NEED TO SPECIFY IN THE COMMAND THE NUMBER OF BYTES PER WORD.

The Return Address Length field is set to zero if logical addressing is being used. If path addressing is being used then the Return Address Length field has to be set to the smallest number of 32-bit words that can be used to contain the path address from the

destination node that is being written to back to the source of the command packet. For example, if three path address bytes are required then the Return Address Length field is set to one.

The reserved Options byte should be set to zero.

The Extra Return Address bytes contain any required path address bytes needed to route the reply packet from the destination node back to the source node. If logical addressing is being used then the Extra Return Address bytes are not present.

The Return Address byte contains the logical address of the source of the write command packet. If the source node does not have a logical address because only path addressing is being used then the Return Address byte must be set to 254 (0FEh) which is the default logical address.

The Transaction Identifier bytes are set to the next transaction identifier in the sequence held by the source node. This uniquely identifies the transaction being started by the write command. The reply to the write command will contain the same transaction identifier and can thus be readily matched to the specific command that caused the reply.

The four Write Address bytes hold the 32-bit memory/register address to which the data in a write command is to be written. The first byte sent in the command is the most significant byte of the address.

The three Data Length bytes contain the length of the data that is to be written. If a single byte is being written this field is set to one. If set to zero then no bytes will be written to memory which may be used as a test transaction. The first byte sent is the most significant byte of the data length.

The Header Checksum byte is an 8-bit checksum used to confirm that the header is correct before executing the command.

The Data bytes contain the data that is to be written into the memory of the destination node. When writing to memory organised in words (e.g. 32-bit words) then the first byte sent is the most-significant byte of the word.

The Data Checksum byte contains an 8-bit checksum used to confirm that the data was correctly transferred. In a write command data is written to target memory provided that the header checksum shows no error in the header. This helps to prevent inadvertent writing to incorrect areas of memory when there is an error in the header. If there is an error in the data checksum then the wrong data will have been written to memory, but is will not have been written to the wrong place. The user application at both source and destination will be informed that there was an error in the data transferred so that corrective action can be taken.

EOP character is the End Of Packet market of the SpaceWire packet.

6.3.2 Write reply

The reply to a write command is sent by the destination back to source of the write command. The reply is used to indicate the success or failure of the write command. The format of the write reply is shown in Figure 2.

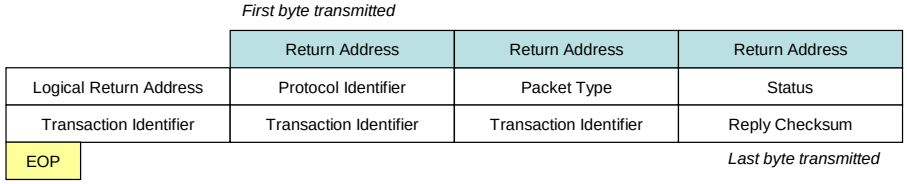

Figure 2 Write Reply Format

The Return Address bytes contain any required path address bytes needed to route the reply packet from the destination node back to the source node. The value of the Return Address bytes are as specified in the Extra Return Address field of the write command. If logical addressing is being used then the Return Address bytes are not present in the reply to the write command. Any Return Address bytes are stripped off by the time the reply reaches the source of the write command.

The Logical Return Address byte contains the logical address of the source of the write command packet, as specified in the write command Return Address field.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is clear (0) to indicate that this is a reply packet.

The Command, Command Options and Return Address Length field are set to the same values as in the command byte of the write command.

The Status byte provides the status of the write command. This is set to zero if the command executed successfully and to a non zero error code if there was an error.

The Transaction Identifier bytes are set to the same value as provided in the write command. This is so that the source of the write command can associate the reply with the original write command.

The Reply Checksum byte is an 8-bit checksum used to confirm that the reply packet has been received without error.

6.3.3 Write action

The operation of the write command is illustrated in the sequence diagram of Figure 3.

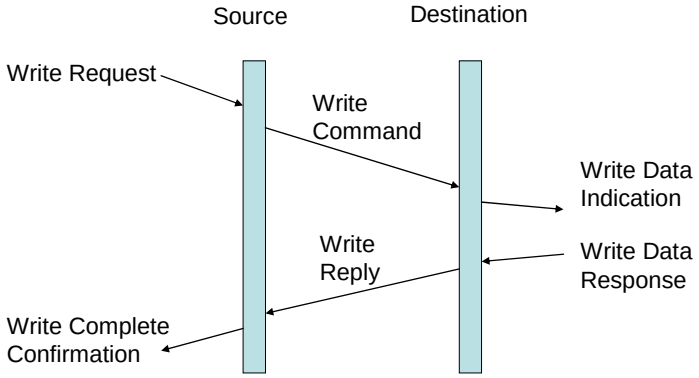

Figure 3 Write Command/Acknowledge Sequence

The write command sequence begins when an application requests to perform a write operation (Write Request). In response to this the source node builds the write command and sends it across the SpaceWire network to the destination node (Write Command). When the Write Command arrives at the destination the header is first checked for errors and if it is ok then the data contained in the write command is written into the specified memory location of the destination node. Once this has been done the user application running on the destination node is informed that a write operation has taken place (Write Data Indication). The user application acknowledges this (Write Data Response) and the destination node then sends a write reply packet back to the source of the write command

(Write Acknowledge). When the write reply is received, the source node indicates successful completion of the write request (Write Complete Confirmation).

NOTE DO WE WANT THE DESTINATION NODE TO FIRST AUTHENTICATE OR ACCEPT THE WRITE BEFORE IT TAKES PLACE. THIS WILL SLOW THINGS DOWN? IT COULD BE A OPTION? PROBABLY BETTER TO ASSUME THAT THE SOURCE KNOWS WHAT IT IS DOING AS THIS IS SUPPOSED TO BE A LOW-LEVEL SERVICE.

6.3.4 Write Errors

There are several error conditions that can arise during a write operation: Write Command Header Error, Write Command Data Error and Write Reply Error.

The sequence of events that occurs when there is an error in the header of the write command is illustrated in Figure 4.

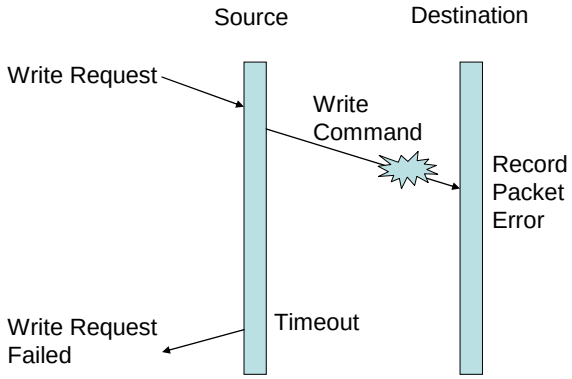

Figure 4 Write Command Header Error

The Write Command packet arrives at the destination and its header is found to be in error. This fact is added to the error statistics in the destination node. The remainder of the packet is discarded. No other action is taken at the destination node, specifically no data is written into the memory of the destination node and no write reply packet is sent back to the source node. Since the source node does not receive a write reply packet, it will eventually timeout or otherwise note that no write reply packet has been received with the transaction identifier of the write command. The error is then flagged to the application that made the write request (Write Request Failed).

The situation that arises when there is an error in the data field of the write command is shown in Figure 5.

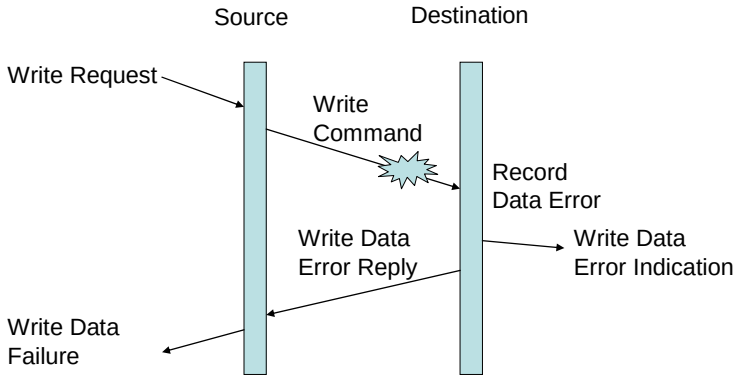

Figure 5 Write Command Data Error

Since the header of the write command has been received without error, the destination node starts to transfer data from the data field of the received packet into destination node memory. If there is insufficient data in the data field (i.e. the data field is shorter than the data length provided in the write command header) then when the EOP is reached data will stop being transferred into destination memory and an error flag will be raised. If there is too much data in the data field then the specified amount of data, defined by the data length field of the write command header, will be transferred to memory, the rest of the packet will be discarded and an error flag will be raised. If there is a data checksum error then an error flag will be raised after the data has been transferred to destination memory. These various errors will be reported to the user application running on the destination node (Write Error Indication). Since the header of the write command was intact it is possible to report the error back to the source. A write reply packet is sent back to the source node indicating the type of error that has occurred (Write Data Error Reply). When this is received at the source node the error is reported to the application that requested the write command (Write Data Failure).

It is possible that the write reply is corrupted or for some other reason does not reach the source node intact. This situation is illustrated in Figure 6.

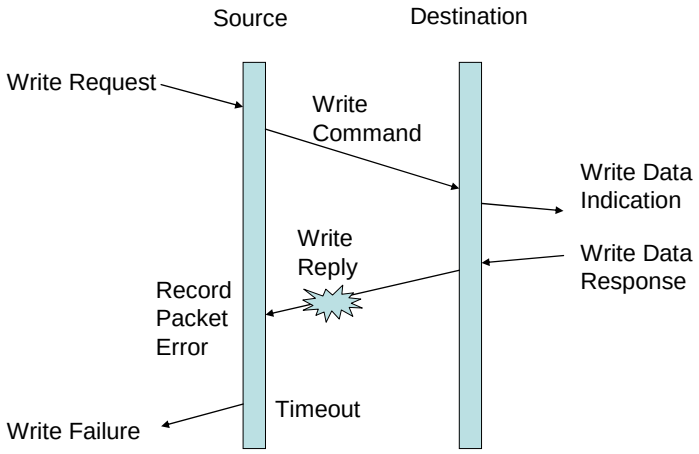

Figure 6 Write Reply Error

The data has been correctly written into destination memory and the destination application has been informed. The write reply that is sent back to the source node is corrupted. Since a correct write reply does not arrive at the source node, the source times out and flags an error to the application that requested the write operation (Write Failure).

6.3.5 Write Request Parameters

The Write Request has to provide the following parameters:

- Destination address
- Return address
- Write command options
- Write address
- Data length
- Data

6.4 Direct Read

6.4.1 Read command

The read command provides a means for one node, the source node, to read one or more bytes of data from the memory of a destination node. The format of the command is shown in Figure 7.

First byte transmitted

	Destination Address	Destination Address	Destination Address
Logical Address	Protocol Identifier	Packet Type, Command Return Address Extra Length	Options
Extra Return Address	Extra Return Address	Extra Return Address	Extra Return Address
Return Logical Address	Transaction Identifier	Transaction Identifier	Transaction Identifier
Read Address	Read Address	Read Address	Read Address
Data Length	Data Length	Data Length	Header Checksum
EOP			*Last byte transmitted*

Bits in Packet Type / Command / Return Address Extra Length Byte

Command/ Response	Read/ Write	Direct/ Indirect	Increment/ No inc. source	Increment/ No inc. target	Ack/ No ack	Extra Return Addr Words	Extra Return Addr Words

Figure 7 Read Command Format

The Destination Address is the address on the SpaceWire network of the node from which data is to be read. The destination address is made up of two parts: the Destination Address bytes which are optional (shaded in Figure 1) and the Logical Address. If path addressing is being used then the Destination Address bytes contain the path to the destination node. The Logical Address is byte is then set to the logical address of the destination node or to the default value 254 (0FEh). If logical addressing is being used there are no Destination Address bytes and the logical address is set to the logical address of the destination node. Normally logical addressing would be used and there would be no Destination Address bytes.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is set (1) indicate that the packet is a command packet, rather than a reply packet.

The Command field hold the direct read command.

The command option “Increment / No Increment Source” is not used by the write command and should be set to zero.

The command option “Increment / No Increment Target” is used for multiple data byte transfers. If set (1) it causes the read address in the target to increment on every byte (or word as determined by the target unit) so that data bytes are read from consecutive memory locations. If not set (0) the read address is not incremented so successive data bytes (or words as determined by the target unit) are read from the same memory location. Note that the width of the memory word is determined by the target unit and can be any multiple of 8-bits. For example, if the width of the target unit memory word is 32-bits then four data bytes from the data field of the command are read from one memory location in the target unit.

NOTE DO WE WANT TO ALLOW PADDING OF 8-BITS INTO 16-BIT ETC FIELD? IF SO WE NEED TO SPECIFY IN THE COMMAND THE NUMBER OF BYTES PER WORD.

The Return Address Length field is set to zero if logical addressing is being used. If path addressing is being used then the Return Address Length field has to be set to the smallest number of 32-bit words that can be used to contain the path address from the destination node that is being written to back to the source of the command packet. For example, if three path address bytes are required then the Return Address Length field is set to one.

The reserved Options byte should be set to zero.

The Extra Return Address bytes contain any required path address bytes needed to route the reply packet from the destination node back to the source node. If logical addressing is being used then the Extra Return Address bytes are not present.

The Return Address byte contains the logical address of the source of the read command packet. If the source node does not have a logical address because only path addressing is being used then the Return Address byte must be set to 254 (0FEh) which is the default logical address.

The Transaction Identifier bytes are set to the next transaction identifier in the sequence held by the source node. This uniquely identifies the transaction being started by the read command. The reply to the read command will contain the same transaction identifier and can thus be readily matched to the specific command that caused the reply.

The four Read Address bytes hold the 32-bit memory/register address from which data is to be read. The first byte sent in the command is the most significant byte of the address.

The three Data Length bytes contain the length, in bytes, of the data that is to be read. If a single byte is to be read this field is set to one. If set to zero then no bytes will be read from memory which may be used as a test transaction. The first byte sent is the most significant byte of the data length.

The Header Checksum byte is an 8-bit checksum used to confirm that the header is correct before executing the command.

EOP character is the End Of Packet market of the SpaceWire packet.

6.4.2 Read reply

The read reply contains either the data that was read from the destination node, or an error code indicating why data could not be read. The reply to a read command is sent by the destination node back to the source of the read command. The format of the read reply is illustrated in Figure 8.

First byte transmitted

	Return Address	Return Address	Return Address
Logical Return Address	Protocol Identifier	Packet Type	Status
Transaction Identifier	Transaction Identifier	Transaction Identifier	Pad
Data Length	Data Length	Data Length	Header Checksum
Data	Data	Data	Data
Data	Data	Data	Data
Data	Data Checksum	EOP	

Last byte transmitted

Figure 8 Read Reply Format

The Return Address bytes contain any required path address bytes needed to route the reply packet from the destination node back to the source node. The value of the Return Address bytes are as specified in the Extra Return Address field of the write command. If logical addressing is being used then the Return Address bytes are not present in the reply to the write command. Any Return Address bytes are stripped off by the time the reply reaches the source of the write command.

The Logical Return Address byte contains the logical address of the source of the read command packet, as specified in the read command Return Address field.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is clear (0) to indicate that this is a reply packet.

The Command, Command Options and Return Address Length field are set to the same values as in the command byte of the read command.

The Status byte provides the status of the read command. This is set to zero if the command executed successfully and to a non zero error code if there was an error.

The Transaction Identifier bytes are set to the same value as provided in the read command. This is so that the source of the read command can associate the reply and data in the reply with the original read command.

The three Data Length bytes contain the length, in bytes, of the data that is to be read and returned in the reply packet. The first byte sent is the most significant byte of the data length.

The Header Checksum byte is an 8-bit checksum used to confirm that the header of the reply packet has been received without error.

The Data bytes contain the data that has been read from the memory of the destination node. When reading from memory organised in words (e.g. 32-bit words) then the first byte sent is the most-significant byte of the word.

The Data Checksum byte contains an 8-bit checksum used to confirm that the data was correctly transferred.

EOP character is the End Of Packet market of the SpaceWire packet.

6.4.3 Read action

The operation of the read command is illustrated in the sequence diagram of Figure 9.

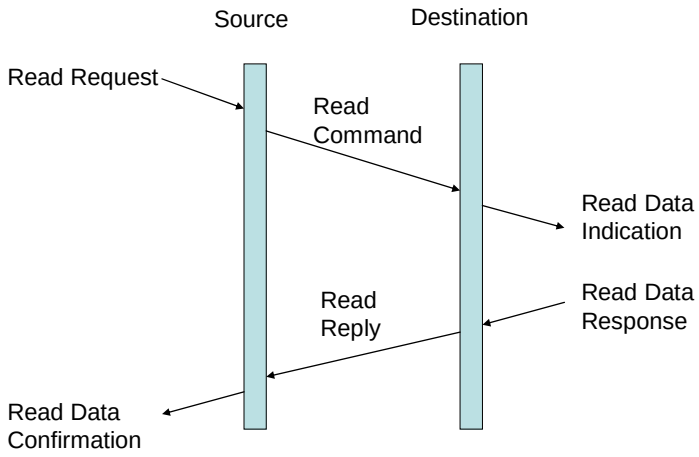

Figure 9 Read Command/Reply Sequence

The read command sequence starts when an application requests to perform a read operation (Read Request). The read command is constructed and sent to the destination node (Read Command). When the read command arrives at the destination it is flagged to the user application on the destination node (Read Data Indication). The header of the read reply packet is formed and the requested data appended to it. The read reply containing the data is then sent back to the source of the read command. When it arrives there the user application that requested the data is informed (Read Data Confirmation).

6.4.4 Read Errors

There are three types of error that can occur when executing a read command: read command error, read reply header error and read reply data error. These errors will now be considered.

The sequence of events following a read command error are illustrated in Figure 10.

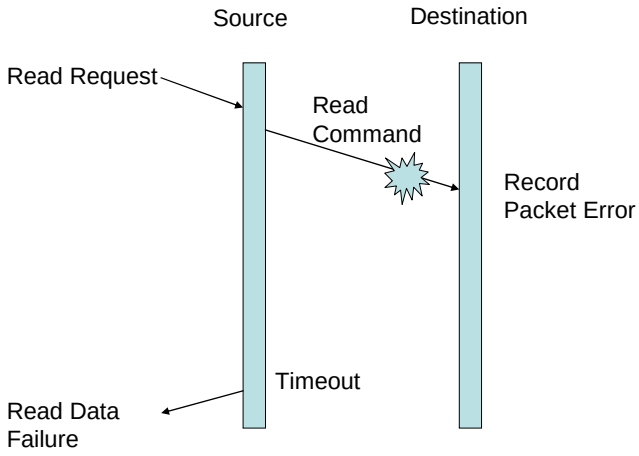

Figure 10 Read Command Error

If the read command is corrupted but arrives at the destination node then a packet error will be recorded at the destination, but no other action will be taken by the destination node. It will not read any data and will not return a read reply packet. If the read command is lost altogether then the destination node would know nothing about the read command at all and would not be able to record a packet error. In either case a timeout will occur at the source node for the read command and the read error will be flagged to the application that sent the command (Read Data Failure).

The situation that arises following a read reply header error is shown in Figure 11.

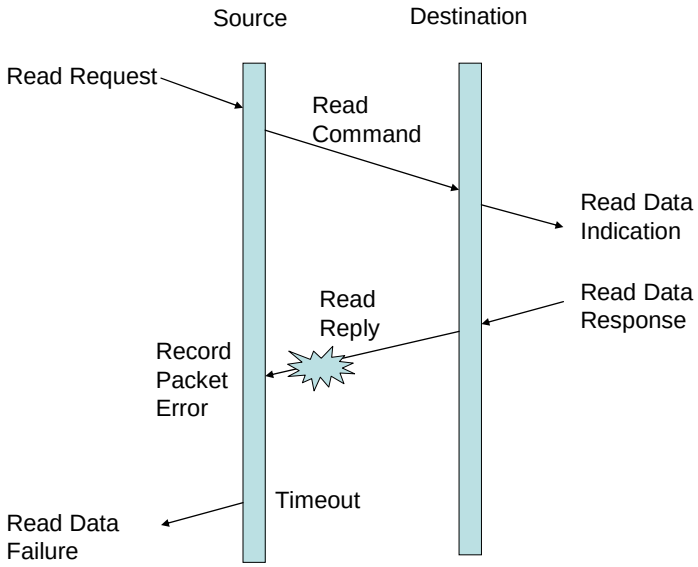

Figure 11 Read Reply Header Error

The read command is received by the destination node and a reply returned to the source node containing the requested data. Either the reply packet gets lost altogether or the header of the read reply is received corrupted and a packet error is recorded at the source. Because there is an error in the header it is not known for certain what transaction identifier the reply packet is for, so nothing else can be done. Eventually there will be a timeout at the source node waiting for the reply packet. A Read Data Failure can then be flagged to the application waiting for the result of the read command.

The result of an error in the data field of a read reply is illustrated in Figure 12.

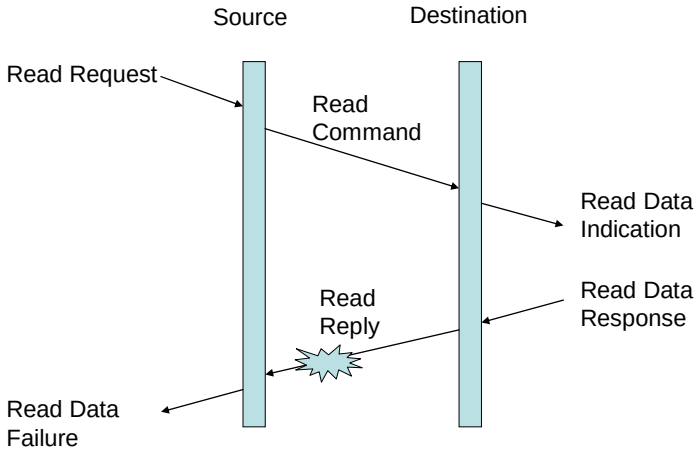

Figure 12 Read Reply Data Error

If the header of the read reply packet is received intact but the data field is corrupted as indicated by an incorrect data field length (too long or too short) or by a checksum error, then an error can be flagged to the application immediately (Read Data Failure) without having to wait for a timeout.

6.4.5 Read Command Parameters

The Read Request has to provide the following parameters:

- Destination address
- Return address
- Read command options
- Read address
- Data length
- Buffer in which to put the received data

6.5 Indirect Write

An Indirect Write Command is a command that asks an intermediate node to send a Write Command to a destination node. When the intermediate node receives the Indirect Write Command it forms a Write Command appending data read from memory in the intermediate node. It then sends this Write Command to the specified destination node which writes the data into the specified position in its memory space and returns a Write Reply back to the intermediate node. The intermediate node then replies to the original source node indicating that the Indirect Write Command has completed.

This type of indirect command is useful when a control node wants to set up data transfers between two other nodes, but does not want to handle the data itself. For example, a control processor could set up a transfer from a camera to a solid state mass memory unit. An image could then be transferred from the camera to memory without the control processor being involved.

6.5.1 Indirect write command

The format of the indirect write command is illustrated in Figure 13

First byte transmitted

	Intermediate Address	Intermediate Address	Intermediate Address
Intermediate Logical Address	Protocol Identifier	Packet Type, Command Target Address Extra Length	Options
Transaction Identifier	Transaction Identifier	Transaction Identifier/Pad?	Pad?
Read Address	Read Address	Read Address	Read Address
Extra Target Address	Extra Target Address	Extra Target Address	Extra Target Address
Target Logical Address	Protocol Identifier	Packet Type, Command Return Address Extra Length	Options
Extra Return Address	Extra Return Address	Extra Return Address	Extra Return Address
Return Logical Address	Pad Space for Trans ID	Pad Space for Trans ID	Pad Space for Trans ID
Write Address	Write Address	Write Address	Write Address
Data Length	Data Length	Data Length	Header Checksum
EOP			

Last byte transmitted

Figure 13 Indirect Write Command Format

The Intermediate Address is the path address (or regional logical address) on the SpaceWire network of the intermediate node on the SpaceWire network that is to perform the write command. The Intermediate Address is not needed if logical addressing is being used. The Intermediate Address bytes will all be stripped off by the time the packet reaches intermediate node.

The Intermediate Logical Address is the logical address of the intermediate node. This byte will be the first byte in the packet when it arrives at the intermediate node and may be used to confirm that the packet has arrived at the correct node. If the path addressing is being used and the intermediate node does not have a logical address then the Intermediate Logical Address byte should be set to 254 (0FEh) which is the default logical address.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is set (1) indicate that the packet is a command packet, rather than a reply packet.

The Command field hold the indirect write command.

The command option “Increment / No Increment Source” is used for multiple data byte transfers from the intermediate node to the target node. Data is read from memory in the intermediate node and then transferred and written to the destination node. If the “Increment / No Increment Source” bit is set (1) is causes the read address in the intermediate node to increment on every byte (or word as determined by the intermediate unit) so that data bytes are read from consecutive memory locations. If not set (0) the read address is not incremented so successive data bytes (or words as determined by the intermediate unit) are read from the same memory location. Note that the width of the memory word is determined by the intermediate unit and can be any multiple of 8-bits. For example, if the width of the intermediate unit memory word is 32-bits then four data bytes from the data field of the command are read from one memory location in the intermediate unit.

NOTE DO WE WANT TO ALLOW PADDING OF 8-BITS INTO 16-BIT ETC FIELD? IF SO WE NEED TO SPECIFY IN THE COMMAND THE NUMBER OF BYTES PER WORD.

The Target Address Extra Length field is set to zero if logical addressing is being used. If path addressing is being used then the Target Address Extra Length field has to be set to the smallest number of 32-bit words that can be used to contain the path address from the intermediate node to the target node for the write command. For example, if three path address bytes are required then the Target Address Extra Length field is set to one.

The reserved Options byte should be set to zero.

The Transaction Identifier bytes are set to the next transaction identifier in the sequence held by the source node. This uniquely identifies the transaction being started by the intermediate write command. The reply to the intermediate write command will contain the same transaction identifier and can thus be readily matched to the specific command that caused the reply. Note that a separate transaction identifier will be used for the exchange between the intermediate node and the target node.

The Pad byte(s) are use for 32-bit alignment of the command fields.

The Read Address bytes are used to specify the start address of the read from memory of the intermediate node.

The remaining fields in the indirect write command form the header of the write command that is to be sent from the intermediate node to the target node.

The Extra Target Address bytes contain any required path address bytes needed to route the write command packet generated by the intermediate node to the target node which is to be written to. If logical addressing is being used then the Extra Return Address bytes are not present.

The Target Logical Address byte contains the logical address of the target of the indirect write command packet. If the target node does not have a logical address because only path addressing is being used then the Target Logical Address byte must be set to 254 (0FEh) which is the default logical address.

The Protocol Identifier byte, Packet Type etc. byte, and Options byte are as specified in the write command (section 6.3.1).

The Extra Return Address bytes are the return path address (or regional logical address) from the target node to the intermediate node.

The Return Logical Address byte holds the logical address of the intermediate node or the default logical address (254). It should have the same value as the Intermediate Logical Address byte.

The Pad Space for Trans ID bytes are set to zero. The transaction identifier for the write transaction between the intermediate node and the target node has to be generated by the intermediate node. It cannot be specified a priori by the source of the indirect write command since this node does not know the current transaction identifier count of the intermediate node. Space is reserved in the command for the transaction identifier so that the intermediate node can simply insert the appropriate transaction identifier.. This uniquely identifies the transaction being started by the write command from the intermediate node. The reply to this write command will contain the same transaction identifier and can thus be readily matched to the specific command that caused the reply.

The four Write Address bytes hold the 32-bit memory/register address to which the data in a write command is to be written. The first byte sent in the command is the most significant byte of the address.

The three Data Length bytes contain the length of the data that is to be written. If a single byte is being written this field is set to one. If set to zero then no bytes will be written to memory which may be used as a test transaction. The first byte sent is the most significant byte of the data length.

The Header Checksum byte is an 8-bit checksum used to confirm that the header is correct before executing the command. The header checksum covers information from the Intermediate Logical Address byte to and including the Data Length bytes. Note that when the write command is forwarded by the intermediate node the header checksum must e recomputed.

EOP character is the End Of Packet market of the SpaceWire packet.

The data bytes for the write command will be appended to the write command sent by the intermediate node to the target node. These data bytes are read from the specified area in memory (Read Address) in the intermediate node and written to the specified area of memory (Write Address) in the target node.

6.5.2 Indirect write reply

First byte transmitted

	Return Address	Return Address	Return Address
Logical Return Address	Protocol Identifier	Packet Type	Status
Transaction Identifier	Transaction Identifier	Transaction Identifier	Pad
Header Checksum	EOP		

Last byte transmitted

Figure 14 Indirect Write Reply Format

The Return Address bytes contain any required path address bytes needed to route the reply packet from the destination node back to the source node. The value of the Return Address bytes are as specified in the Extra Return Address field of the indirect write command. If logical addressing is being used then the Return Address bytes are not

present in the reply to the indirect write command. Any Return Address bytes are stripped off by the time the reply reaches the source of the indirect write command.

The Logical Return Address byte contains the logical address of the source of the indirect write command packet, as specified in the indirect write command Return Address field.

The Protocol Identifier byte is set to the value 1 (01h) TBC which is the Protocol Identifier for the Remote Memory Access protocol.

The Packet Type field is clear (0) to indicate that this is a reply packet.

The Command, Command Options and Return Address Length field are set to the same values as in the command byte of the indirect write command.

The Status byte provides the status of the indirect write command. This is set to zero if the command executed successfully and to a non zero error code if there was an error.

The Transaction Identifier bytes are set to the same value as provided in the indirect write command. This is so that the source of the indirect write command can associate the reply with the original indirect write command.

The Reply Checksum byte is an 8-bit checksum used to confirm that the reply packet has been received without error.

6.5.3 Indirect write action

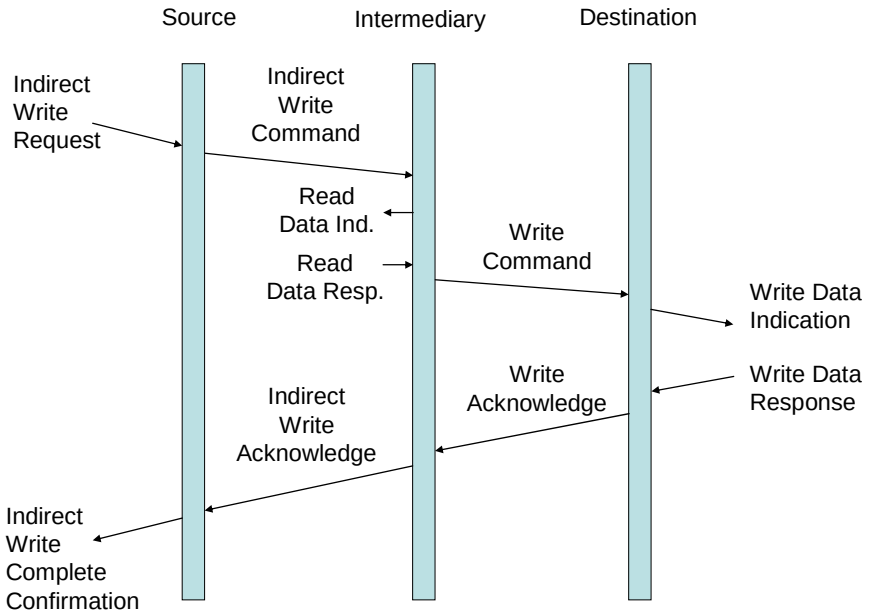

Figure 15 Indirect Write Command/Reply Sequence

6.6 Indirect Read

6.6.1 Indirect read command

First byte transmitted

	Destination Address	Destination Address	Destination Address
Logical Address	Protocol Identifier	Packet Type, Command Target Address Extra Length	Options
Transaction Identifier	Transaction Identifier	Transaction Identifier/Pad?	Pad?
Write Address	Write Address	Write Address	Write Address
Extra Target Address	Extra Target Address	Extra Target Address	Extra Target Address
Target Address	Protocol Identifier	Packet Type, Command Return Address Extra Length	Options
Extra Return Address	Extra Return Address	Extra Return Address	Extra Return Address
Return Address	Pad Space for Trans ID	Pad Space for Trans ID	Pad Space for Trans ID
Read Address	Read Address	Read Address	Read Address
Data Length	Data Length	Data Length	Header Checksum
EOP			

Last byte transmitted

Figure 16 Indirect Read Command Format

6.6.2 Indirect read reply

First byte transmitted

	Return Address	Return Address	Return Address
Logical Return Address	Protocol Identifier	Packet Type	Status
Transaction Identifier	Transaction Identifier	Transaction Identifier	Pad
Header Checksum	EOP		

Last byte transmitted

Figure 17 Indirect Read Reply Format

6.6.3 Indirect read action

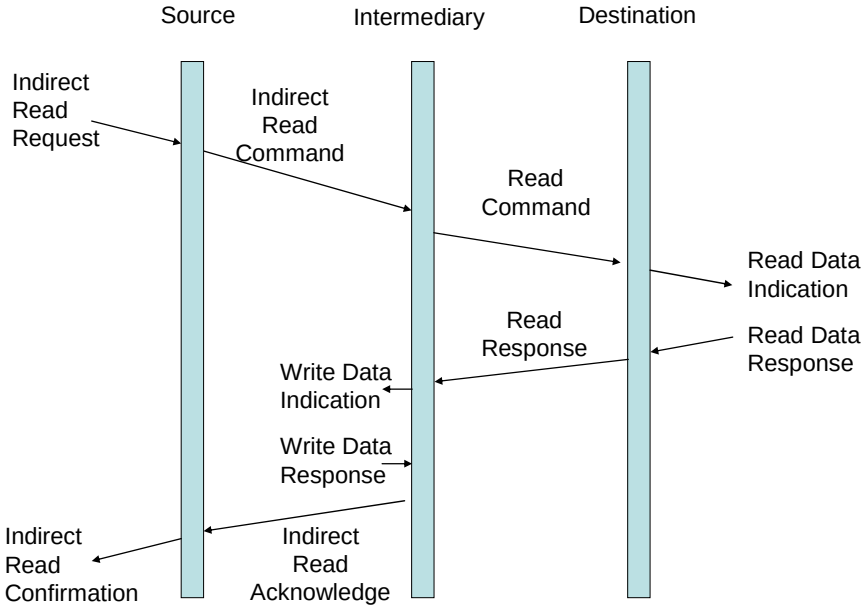

Figure 18 Indirect Read Command/Reply Sequence