High Speed & ASIC Design

Streaming Protocol

Specification

Author(s): Digital Design
Date: September 20, 2019
Version: 2.2 GIT Info: HEAD-b929 9db6584fd

Contents

1 Introduction **5**

2 Infiniband Settings **7**

3 Protocol Description **8**

3.1 Status Channel Protocol 8

3.1.1 Connection Establishment 9

3.1.2 Status Channel Setup 10

3.1.3 Data Channel Setup 11

3.1.4 Channel Termination 11

3.2 Data Channel Protocol 11

3.2.1 Data Transfer 11

3.2.2 Data Reception 12

3.2.3 RDMA write response 12

4 Data mapping **16**

4.1 Frame 16

4.2 Packet 16

4.3 Word 16

4.4 4-bit image 18

4.5 6-bit image 18

4.6 Image CRC 19

List of Figures

1.1 Simplified Communication Overview 5

3.1 Flow Diagram Of The Communication Setup 10
3.2 Memory regions during an exposure. The 64-bit virtual address space of the RDMA write request is mapped into a ring buffer. The write window is defined by the current read pointer and the ring buffer size. 13
3.3 Data stream monitoring and control during an exposure. 14

4.1 packets within a frame 16
4.2 words within a packet 16
4.3 bytes within a word 17
4.4 4-bit image data mapping 18
4.5 6-bit image data mapping 19

List of Tables

3.1 Status Packet Format 9
3.2 Status Channel Methods 9
3.3 RDMA Header 11
3.4 Errors And Acknowledgments 13
3.5 ACK/NACK Packet Format 13
3.6 Acknowledge Types 14
3.7 Event Types 15

1 Introduction

This document describes the *IMS Streaming Protocol*, i.e. the transmission rules between the *Rasterizer Workers* and the *WCU.BUF* FPGA design. The communication is based upon the *Infiniband* protocol, a switched fabric communication link used in high-performance computing [4]. Detailed information about the Infiniband standard can be found in [2] and [1] respectively. The lecture of this document requires understanding of this particular part of data transmission issues, especially the concept of *RDMA* (*Remote Direct Memory Access*). A simplified architectural overview of the participants in the communication process is shown in Figure 1.1.

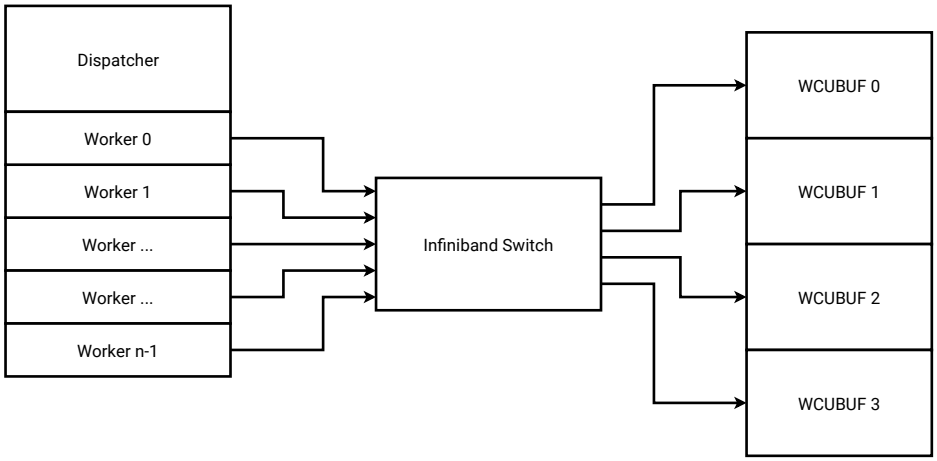

Figure 1.1: Simplified Communication Overview

Every Rasterizer Worker has two logical Infiniband connections to every WCU.BUF unit (indicated by one visible connection in Figure 1.1):

- The **Status Channel** is used for configuration and connection setup and is connected to a dedicated NIOS2 soft-cpu (called the *Infiniband (IB) NIOS system*) implemented in every WCU.BUF unit. It employs the *Unreliable Datagram SEND* service to exchange information.
- The **Data Channel** transports exposure data from a Worker to a WCU.BUF unit as well as the receive acknowledgments in the oppisite direction. The exposure data is transmitted through *Unreliable Connection RDMA WRITE* operations, while *Unreliable Connection SEND* operations carry the response from a WCU.BUF unit.

The Infiniband core features two basic communication paths:

- The *CPU interface* provides a register-based access to the internal settings of the IP. This memory-mapped interface also provides a communication path through FIFO registers (Soft Packet Interface) [3], which are attached to the *Soft Queue Pairs (soft QPs)* of the Infiniband core. All communication with the WCU.BUF IB NIOS CPU needs to proceed through the Soft Queue Pairs.

- Contrary to the memory-mapped approach of the CPU interface the *DMA interface* follows a pipeline pattern. The data arriving in the WCU.BUF through this interface are directly processed and forwarded to the DDR3 memory by the hardware. All communication through the DMA interface comes from the *Hard Queue Pairs (hard QPs)* of the Infiniband core.

The communication structure of the employed Infiniband IP sets the major cornerstones for the design of the streaming protocol. Typical RDMA environments found in Infiniband based computer clusters use either TCP/IP or a service-based connection management for the Status Channel. For the WCU.BUF, however, both of these options have been deemed too complex, as they would result in a considerable development effort on the FPGA side.

The Status Channel uses the soft Queue Pair path as it involves the CPU. The Data Channel communication uses the hard QP path and does not involve any direct actions from the firmware. The Status Channel is implemented in a custom protocol based on Unreliable Connection SEND datagrams, which makes optimal use of the Infiniband core's capabilities. The protocol is implemented on the WCU.BUF side in the IB-NIOS firmware and on the rasterizer side using the low-level *Infiniband Verbs API* implementation provided with the OFED distribution.

2 Infiniband Settings

Due to constraints in the Infiniband IP regarding the handling of soft- and hard-QP traffic, a proper operation of the streaming protocol requires special configuration settings in the infiniband fabric. The configuration of an infiniband subnet is managed by the *Infiniband subnet administration*. In the IMS EMET tool cluster, the subnet administration is carried out by the OpenSMD subnet management daemon which runs as a service on the central file server. The subnet manager sweeps the network for switches, TCAs, and HCAs and configures all entities according to their respective capabilities (e.g. LID, MTU, SL/VL settings). It further provides a database listing information for all entities in the subnet seeking to engage in any form of communication through the fabric.

The WCU.BUF modules use *fixed virtual lane (VL) assignments* for soft and hard queue pairs, respectively. Consequently, virtual lanes can only be configured for either the soft QP traffic (to the CPU) or the hard QP traffic (to the DMA interface), but not both. Therefore, Rasterizer needs to assign special and separate *service levels (SLs)* to the communication pakets for the Status Channel and the Data Channel, respectively. These service levels have to be assigned to the proper virtual lanes in the SL to VL mapping table of the subnet management daemon. On the IMS EMET tool the typical configuration uses SL0 and VL0 for the software path and SL1 and VL1 for the hardware path.

3 Protocol Description

This chapter describes the protocols of the two logical channels between every Rasterizer Worker and every WCU.BUF unit in more detail. The Infiniband RDMA data transfer model serves as the basis for the implementation. The protocol is based on the following premises:

- All communication in the EMET tool involves only one Infiniband subnet.
- Every entity in the Infiniband subnet has a *Local Identifier* (LID) assigned by the subnet manager.
- Different logical communication ports on the same entity use different *Queue Pairs (QPs)*, which are identified by their *Queue Pair Number (QPN)*. These Queue Pairs are the endpoints of every Infiniband based communication.
- To send data to another entity in the Infiniband network, at least the LID and the QPN of the receiver have to be known by the sender.
- For data transfers using the *Unreliable Datagram* (UD) service, QPs on either side are associated with a *Queue Key (QKEY)*. The receiving queue only accepts data packets which carry the correct QKEY in their header.
- For RDMA data transfers (e.g. RDMA Write) on an *Unreliable Connection (UC)*, the receiving queue is associated with a memory region characterized by a *Virtual Address (VA)*, the size of the region and a *Remote Key (RKEY)*. The receiver only accepts data that carries the RKEY and a valid VA (inside that memory region) in the header. Data is written to the memory region without any further notification to the operating system of the receiver (and any higher level protocol if applicable).

The IMS Rasterizer to WCU.BUF streaming protocol adds two minor modifications to the Infiniband UC RDMA data transfer model:

1. If a WCU.BUF module receives a UC RDMA WRITE transmission it responds with an acknowledgment that needs to be received and processed by the sender.
2. The memory on a WCU.BUF module is implemented as a ring buffer and appears to the Rasterizer Workers as a memory spanning the full 64-Bit address space.

3.1 Status Channel Protocol

The status channel is used for the configuration and for the setup of the RDMA channel (Data Channel). On the WCU.BUF side the status channel communication is handled by the Firmware of the IB NIOS CPU. In order to make it possible for the Rasterizer worker to set up the connection through Infiniband, all Status Channel communication happens through a dedicated QP on the WCU.BUF side, whose QPN is known to the Rasterizer workers. Information on the Status Channel is exchanged

Bits		31-24	23-16	15-8	7-0
Bytes	3-0	major version	minor version	reserved	Method
	7-4	LID			
	11-8	QPN			
	15-12	QKEY or QPND			
	19-16	VA (31-0)			
	23-20	VA (63-32)			
	27-24	RKEY			

Table 3.1: Status Packet Format

Method	**Identifier**	**Direction**	**Semantics**
STAT_REQ	0x0	Worker -> WCU.BUF	Status Channel registration request
STAT_RES	0x1	WCU.BUF-> Worker	Status Channel registration response
STAT_TERM	0x2	Worker -> WCU.BUF	Status Channel termination request
STAT_DOWN	0x3	WCU.BUF-> Worker	Status Channel terminated
DATA_REQ	0x4	Worker -> WCU.BUF	Data Channel registration request
DATA_RES	0x5	WCU.BUF-> Worker	Data Channel registration response
DATA_TERM	0x6	Worker -> WCU.BUF	Data Channel termination request
DATA_DOWN	0x7	WCU.BUF-> Worker	Data Channel terminated

Table 3.2: Status Channel Methods

using *Status Packets* which are based on UC SEND datagrams. The Status Packet format is shown in Table 3.1

The WCU.BUF transmits a protocol version number to the rasterizer to indicate supported protocol features. The initial version is 0.0 for compatibility reasons. Introducing end-to-end CRC pushes the version number to 1.0. The compatibility check is performed at the rasterizer side. There is no protocol negotiation whatsoever.

Name vom QPN und QPND sinnvoll ? The Status Channel protocol defines several methods to exchange information between a Rasterizer Worker and a WCU.BUF unit which are defined in Table 3.2. There are four methods for each channel: The REQ and RES methods are used for channel establishment and the TERM and DOWN methods for its termination.

3.1.1 Connection Establishment

The connection setup is initiated by a Rasterizer Worker. In the IMS rasterization datapath, the Rasterizer Workers are provided with the GUIDs of the WCU.BUF cards, instead of their LIDs. Thus, before establishing a connection each Rasterizer Worker obtains the LID of the WCU.BUF unit from the *subnet manager* in a process that is not described in this document. Every WCU.BUF unit uses a fixed QPN and QKEY for the Status Channel and waits for messages. The connection is initialized in two phases:

1. setup of the Status Channel.

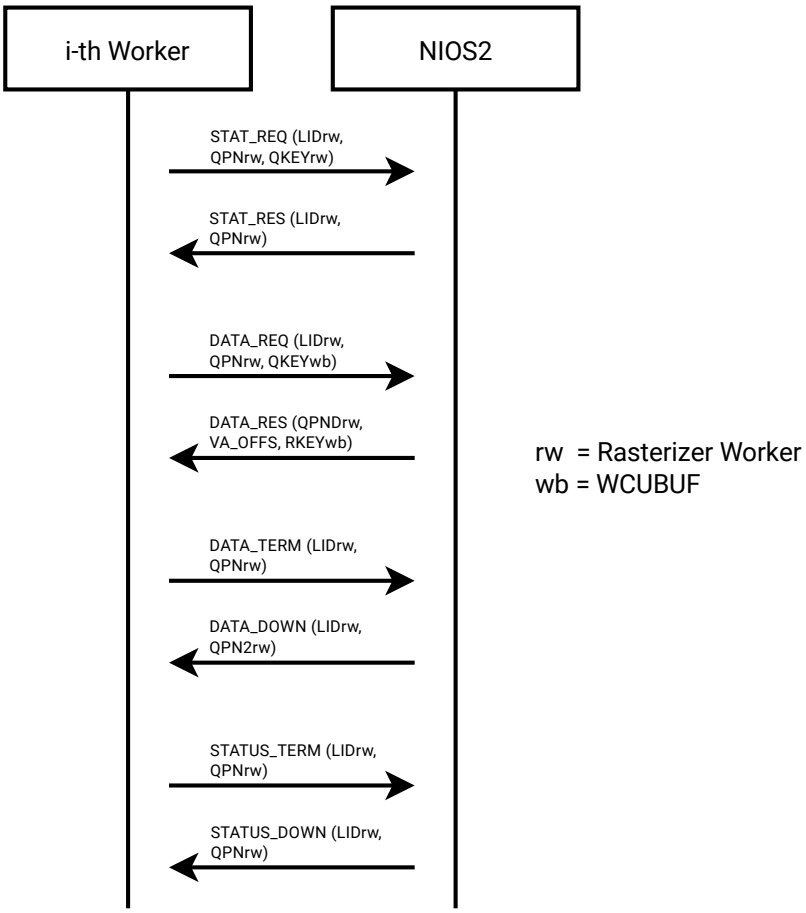

Figure 3.1: Flow Diagram Of The Communication Setup

2. setup of the Data channel.

The flow diagram in Figure 3.1 illustrates the connection setup between Worker i and the WCU.BUF card.

3.1.2 Status Channel Setup

As the WCU.BUF side of the Status Channel does not require initialization (fixed QPN and QKEY), the only purpose of the Status Channel setup is to introduce the Rasterizer Worker to the WCU.BUF unit. To bring up the Status Channel, the LID, the QPN and the QKEY of the RasterizerWorker have to be transmitted (LIDrw, QPNrw, QKEYrw). This is done by sending a STAT_REQ message carrying these parameters as payload. The other parameter fields in the message are ignored by the WCU.BUF firmware. The WCU.BUF module stores the parameters in a table. The combination of LIDrw and QPNrw is used to identify the Rasterizer Worker, and in addition the LIDrw, QPNrw and QKEYrw are needed to compose the reply on the status channel. The NIOS Software replies to STAT_REQ requests with STAT_RES messages. The STAT_RES response holds the same LIDrw and QPNrw parameters, while all other parameter fields have arbitrary values.

Bits		31-24	23-16	15-8	7-0
Bytes	0-3	Virtual Address (63-32)			
	4-7	Virtual Address (31-0)			
	8-11	RKEY			
	12-15	DMA Length			

Table 3.3: RDMA Header

3.1.3 Data Channel Setup

After the Status Channel is set up, the Rasterizer Worker can establish the Data Channel using the DATA_REQ message. The request message needs to carry the LIDrw and the QPNrw of the Status channel in order for the WCU.BUF unit to identify the Worker. In addition, the QPND field is set to the QPNrw of the Data Channel, where RDMA messages are sent and acknowledgments are received. All other parameters are ignored by the WCU.BUF firmware. The transmitted LIDrw and QPNDrw are used to setup the hard QP on the WCU.BUF side. When the QP setup is finished, the WCU.BUF firmware responds by sending a DATA_RES message including the QPNDrw for the Data Channel in the QPND field, the RKEYwb and the VA_OFFS. The LID and QPN of the status channel are again left unchanged for idenfication purposes.

3.1.4 Channel Termination

Channel termination is done for each channel separately. However, as the status channel is needed for the termination of the data channel, the status channel termination also automatically terminates the other channels if they are still open. To terminate the data channel, the Rasterizer Worker sends a DATA_TERM message through the status channel. Again, the LID and QPN parameters are used for identification of the Worker and all other parameters are ignored by the WCU.BUF Firmware. Upon reception of a DATA_TERM packet, the WCU.BUF module frees the resources of the data channel and returns a DATA_DOWN packet with the proper identification fields, LIDrw and QPNrw, and all other fields set to arbitrary values. The status channel termination uses STATUS_TERM and STATUS_DOWN packets, which are similar in nature. When the WCU.BUF module receives a STATUS_TERM package, it removes the respective worker from its worker identification table.

3.2 Data Channel Protocol

Exposure data is transmitted on the Data Channel from a Worker to a WCU.BUF module. As the communication is based on unreliable packets, an acknowledgment mechanism is included into the protocol in order to identify and correct for missing or erroneous data frames by retransmission.

3.2.1 Data Transfer

The streaming protocol is based on Unreliable Connection RDMA WRITE messages sent from the Rasterizer Worker to the WCU.BUF unit. The essential information for the write process is contained in the RDMA packet header, see Table 3.3.

A Worker starts streaming data at the VA returned from the Data Channel setup and sends frames of exposure data of a fixed size (e.g. 32MB). These frames are split by the Infiniband architecture to packets of MTU size and are sent consecutively where only the first packet carries the RDMA header.

The payload size of a packet can vary between 64 bytes to 4096 bytes in steps of 64 bytes. Packets with a payload less than 64 bytes can not be properly handled and are discarded. Every frame has to start at a 64 byte aligned address. A frame typically consists of packets of the same size except for the last packet, if the framesize is no multiple of the packet size. Sending more and more frames, the Rasterizer Worker increases the VA until the full mask has been transmitted. The WCU.BUF DMA hardware translates the VAs into internal ring buffer addresses. Writing of data is only allowed in a certain memory region, which is defined by the current exposure data read pointer, see Figure 3.2.

3.2.2 Data Reception

In addition to providing high-speed access to the buffer memory to both the Rasterizer and the TROM Receiver, it is a central task of the WCU.BUF cards to maintain the integrity of the exposure data. In an exposure process, the pixel data written by the Rasterizer workers starts at VA=`0x0000000000000000` and is written substripe-by-substripe consecutively into memory[1]. The Rasterizer Workers continue streaming data into the buffer as long as their data is acknowledged by the WCU.BUF.

The memory range in which write operations are acceptable is called the “write window”. This memory range is defined by the In the current implementation, this window starts at the current read pointer and spans the width of the ring buffer, typically 256GiB minus one frame size, typically 32MiB see Figure 3.2. Contrary to earlier implementations, the write window is defined in the *virtual address space*, rather than in the DDR3 address space. The advantage of this approach is that the upper and lower limits of the write window are always in strict order and can be tested by a simple comparison operation. In the current setup , the enforcement of the window boundaries is shared between the WCU.BUF and the software control. The WCU.BUF enforces the upper boundary of the memory window by holding back the Rasterizer Workers with NACK responses. The software control, as depicted in Figure 3.3, keeps track of the exposable data, i.e. consecutively delivered, substripes. It also makes sure that the data of a stripe is completely delivered to the buffer before it is exposed and that no data is written to the memory region that is currently exposed.

3.2.3 RDMA write response

The streaming protocol compensates for the missing acknowledgement mechanism in the unreliable connection to which the employed Infiniband Core is limited. Thus, the protocol defines an error detection and recovery mechanism on its own, that is implemented in hardware on the WCU.BUF side.

The protocol defines acknowledgment (ACK) and negative acknowledgment (NACK) responses to write requests from Rasterizer Workers. ACK indicates that a frame was successfully received, while NACK indicates an error. Table 3.4 summarizes the different responses as seen by the Rasterizer Worker and the actions to be taken. The timeout is configurable, the default timeout is 20sec. Useful values for timeout and wait period still to be defined!

Acknowledgments are sent by the WCU.BUF modules using UC SEND frames. The ACK/NACK package includes the 64-bit VA of the frame it ACks or NACKs. This should enable a proper relation to the respective frame. However the VA is set to `0x0000000000000000` in case of a No Start Of Frame error. The corresponding data format is shown in Table 3.5. The values for the different acknowledgment and event types are shown in Tables 3.6 and 3.7. For each detected error the corresponding bit is set to 1, otherwise it is set to 0.

[1] Isn’t it amazing that the whole pixel data of one mask will fit into the 64 bit address space?

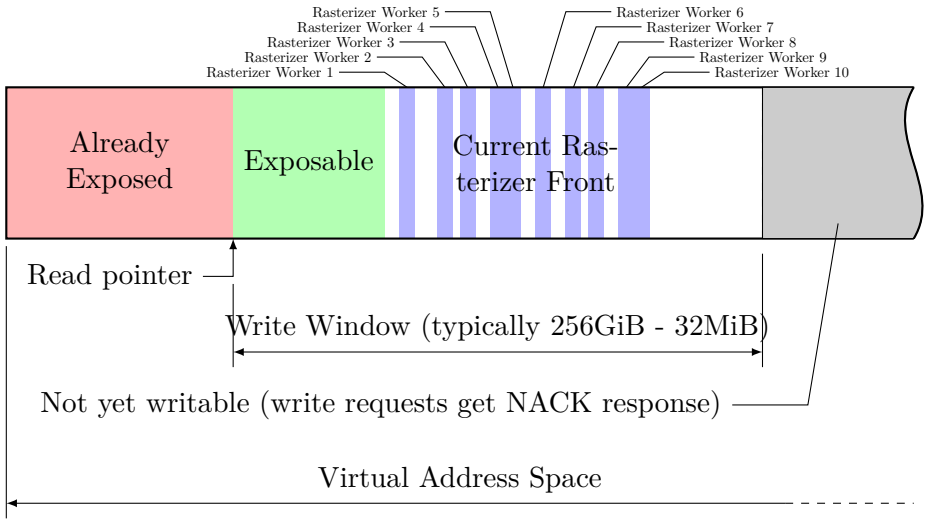

Figure 3.2:
Memory regions during an exposure. The 64-bit virtual address space of the RDMA write request is mapped into a ring buffer. The write window is defined by the current read pointer and the ring buffer size.

Event	**ACK Type**	**Behaviour (Sender)**
Valid frame reveived	ACK	Report successful delivery
Receive Errors (out of sequence, EBF, EBP, parity, corr. ECC, uncorr. ECC, frame/packet length, no SOF)	NACK	Retransmit frame
Frame outside receive window	NACK	Retransmit frame after wait period
Invalid RKEY	NACK	Report error and terminate
Invalid VA	NACK	Report error and terminate
Loss of frame	*timeout*	Retransmit frame
Loss of ACK/NACK	*timeout*	Retransmit frame

Table 3.4: Errors And Acknowledgments

Bits		**7-0**	**15-8**	**23-16**	**31-24**
Bytes	0-3	ACK/NACK Type			
	4-7	Event(s)			
	8-11	Virtual Address (31-0)			
	12-15	Virtual Address (63-32)			

Table 3.5: ACK/NACK Packet Format

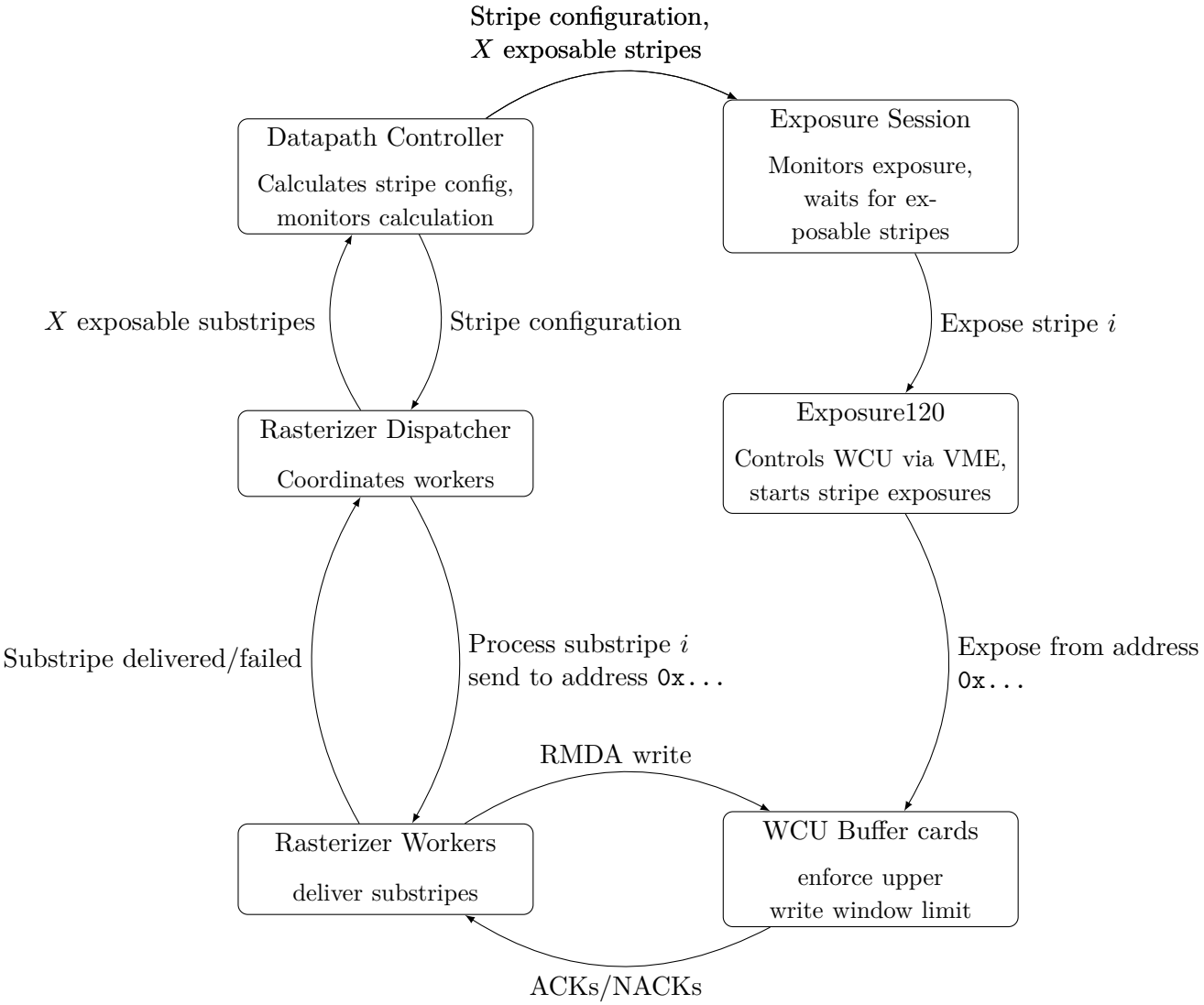

Figure 3.3: Data stream monitoring and control during an exposure.

ACK Type	Value
ACK	0
NACK	1

Table 3.6: Acknowledge Types

Bit	Event(s)	Description
0	Out of sequence error	a packet with an unexpected sequence number was received
1	Parity error	parity check failed at the Josh core IP - IB_DMA interface
2	Outside window error	the virtual address for this frame is currently not mapped to the DDR3 memory
3	Invalid RKEY	the RKEY does not match the key negotiated at connection setup
4	Frame length error	the received bytes differ from the number in the frame header
5	Packet length error	the received bytes differ from the number in the packet header or the packet has less than 64 bytes of payload
6	End of Bad Frame error	EBF received (used by the sender to mark a frame as invalid)
7	End of Bad Packet error	EBP received (used by the sender to mark a packet as invalid. This will result in a bad frame)
8	Invalid VA error	the virtual address is not 64 bytes aligned
9	No Start Of Frame error	the receiver (WCU.BUF) has not seen the SOF for the current frame (it probably was reset in the middle of the frame)
10	correctable ECC error	ECC check at the Josh core IP - IB_DMA interface shows a correctable (single-) bit error
11	uncorrectable ECC error	ECC check at the Josh core IP - IB_DMA interface shows a uncorrectable (multi-) bit error
12-31	-	not used, set to 0

Table 3.7: Event Types

4 Data mapping

This section describes how image data is packed into an Infinband frame.

4.1 Frame

Figure 4.1 shows how packets are organized within a frame.

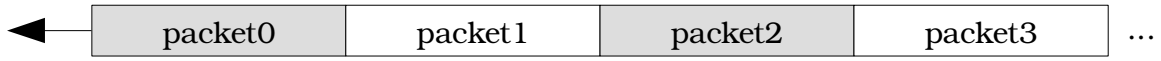

Figure 4.1: packets within a frame

4.2 Packet

Figure 4.2 shows how words are organized within a packet. A packet can hold up to 4096 bytes of payload data according to the Josh core specification[3]. A limitation of the WCU.BUF implementation is a minimum chunk size of 64 bytes. As a result valid packet sizes are 64 bytes to 4096 bytes in steps of 64 bytes. A word at the Josh core interface is 128 bits wide. A minimum sized packet consists of 4 words on the Josh core interface and the maximum sized packet contains 256 words.

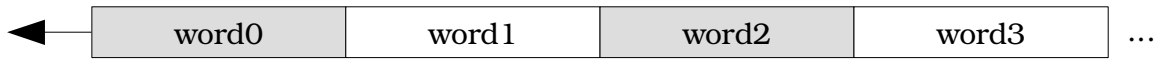

Figure 4.2: words within a packet

4.3 Word

Figure 4.3 shows the byte order within a word received by the Josh core. The byte order is big endian like on most network protocols, whereas the bit order is little endian.

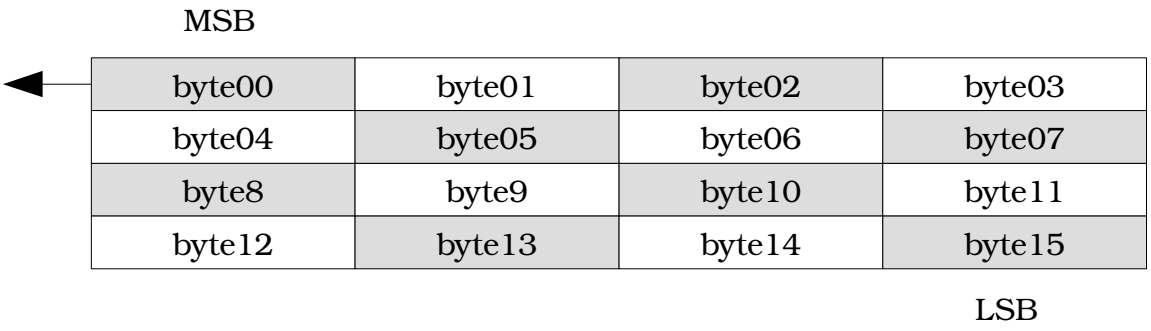

Figure 4.3: bytes within a word

4.4 4-bit image

A WCU.BUF module only handles one quarter of an image. Therefore it only receives a quarter of an image. Figure 4.4 shows how this image data is mapped to the byte stream transferred via Infiniband. There is an image CRC at the beginning followed by a padding area, the VA of the current image followed by another padding area. Subsequently the image data is transferrred column wise using one byte for 2 pixels respectively.

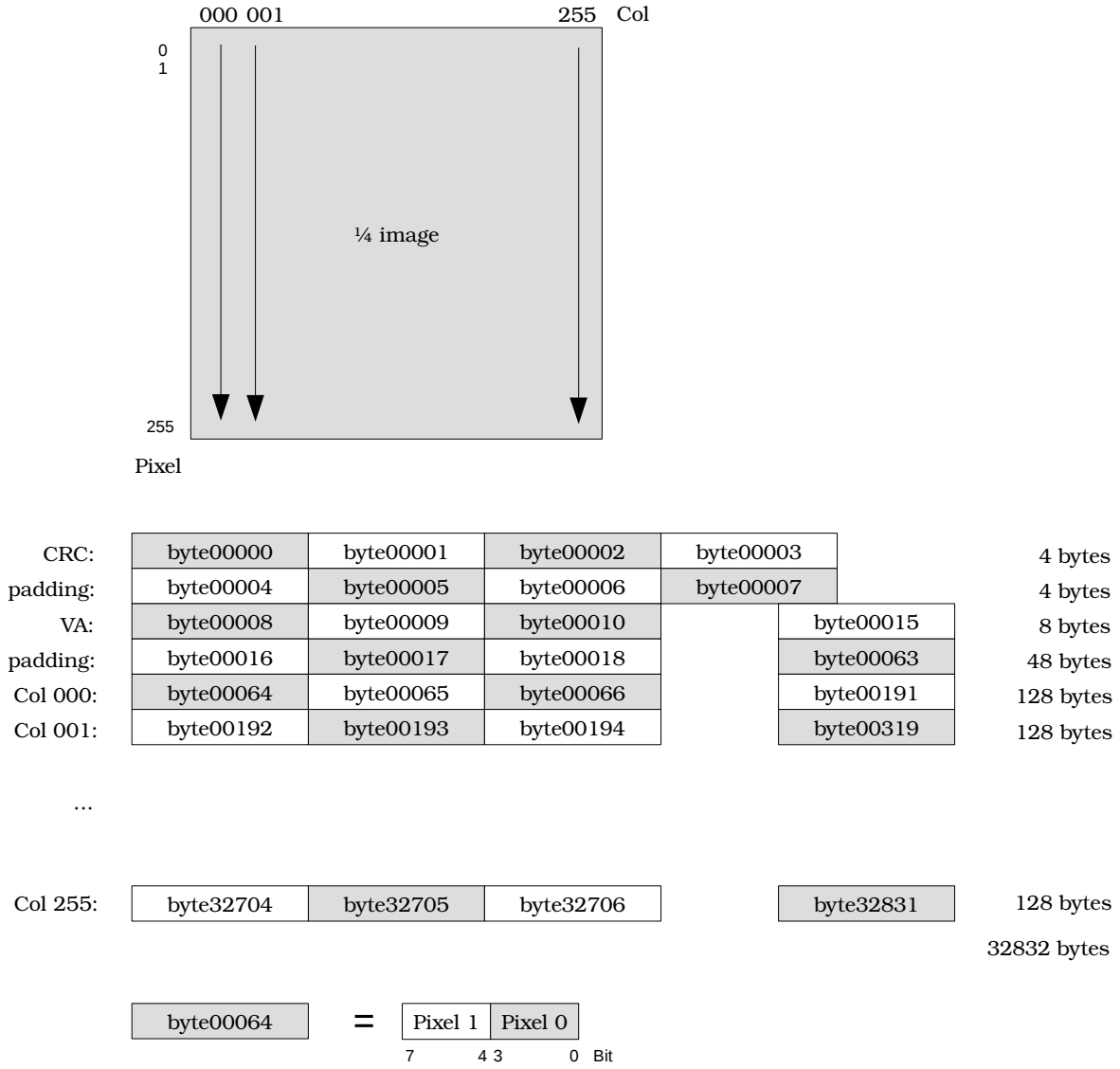

Figure 4.4: 4-bit image data mapping

4.5 6-bit image

Figure 4.5 shows the same for a 6-bit image. The image data is mapped to one pixel per byte. So in pricipal this format can also be used for 8-bit images in the future.

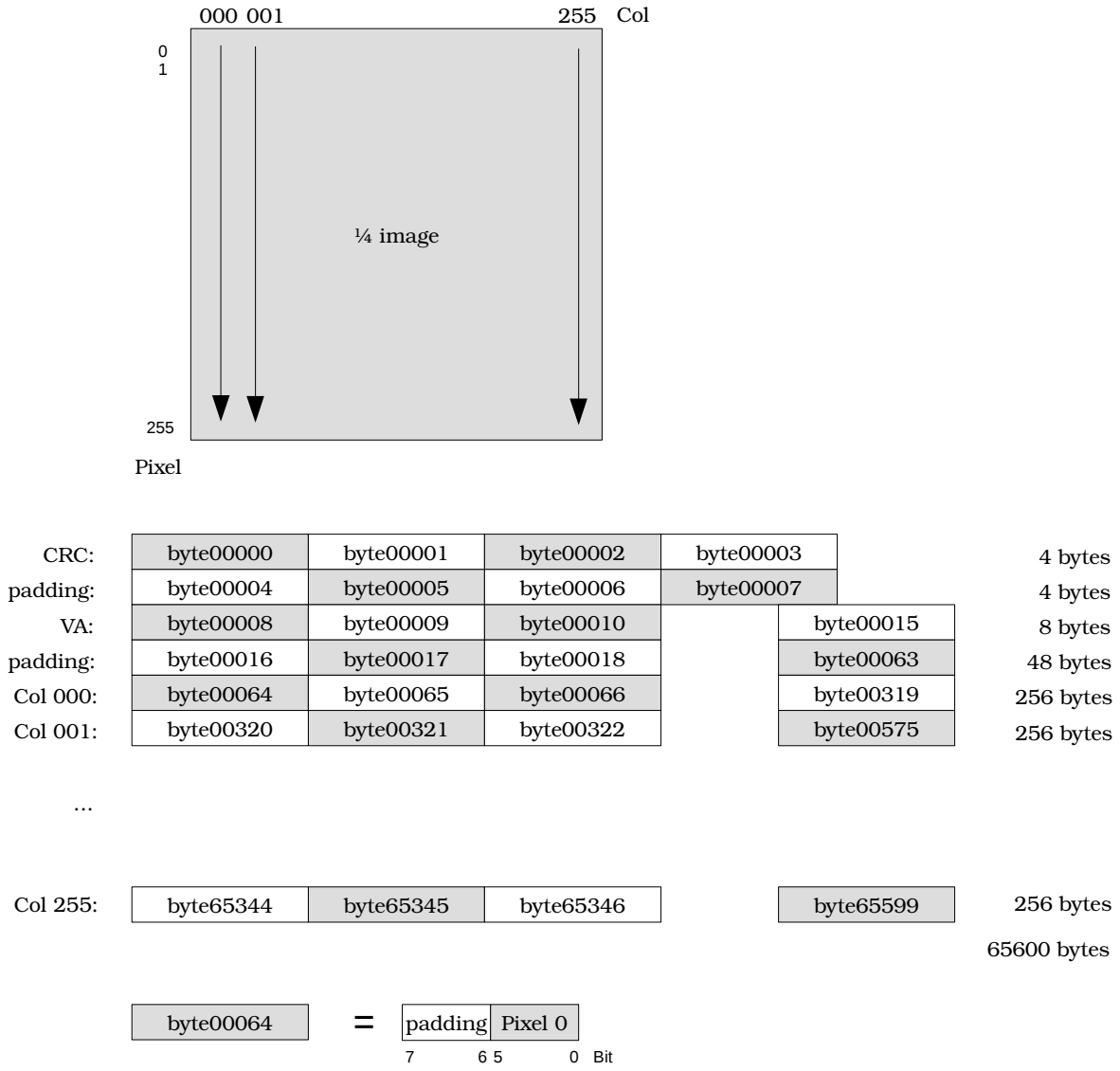

Figure 4.5: 6-bit image data mapping

4.6 Image CRC

The image CRC32 is calculated using the Castagnoli polynomial:

$$crc[31:0] = 1+x^6+x^8+x^9+x^{10}+x^{11}+x^{13}+x^{14}+x^{18}+x^{19}+x^{20}+x^{22}+x^{23}+x^{25}+x^{26}+x^{27}+x^{28}+x^{32}$$

Bibliography

[1] Infiniband Trade Association. *Infiniband Architecture Specification Volume 2*, 1.2.1 edition, 2006.

[2] Infiniband Trade Association. *Infiniband Architecture Specification Volume 1*, 1.2.1 edition, 2007.

[3] B. Joshua Rosen. *Asynchronous Quad Data Rate InfiniBand Link Layer Core Specification*, 2.8 edition, 2006.

[4] Wikipedia. Infiniband — Wikipedia, the free encyclopedia, 2013. [Online; accessed 13-January-2013].

Revison History

Revision	Date	Author(s)	Protocol Rev.	Description
1.0	01/10/2013	MiB	0.0	Initial release
1.1	02/20/2013	MW	0.0	Added use case chapter, status channel and ACK channel description
1.1	03/06/2013	MW	0.0	Added connection setup, acknowledgements and changed RDMA data transport. Added metadata channel.
1.1	07/10/2013	NT	0.0	Added connection setup flow diagram.
1.2	09/27/2013	MW	0.0	Added Infiniband settings and ACK formats.
1.3	02/21/2014	MW	0.0	Changed status channel methods. Added padding to acknowledgement packet.
1.4	03/10/2016	FS	0.0	Updated to current state of implementation. Removed Metadata channel.
1.5	04/01/2016	WE	1.0	Removed Metadata channel. Introduced version number in status packet added 64-bit virtual address in ACK/NACK packages
2.0	07/12/2016	WE	1.0	converted document to lyx and ported to new template
2.0	07/13/2016	WE	1.0	added Data mapping section, overhauled the whole document for consistency, refined channel setup mechanism
2.0	11/15/2016	WE	1.0	add CRC polynomial for image CRC32 calculation
2.0	02/03/2017	WE	1.0	rework NACK handling and event types
2.1	05/29/2017	WE	1.1	change NACK event encoding to be able to report more than 1 error
2.1	05/29/2017	WE	1.1	add out of sequence NACK event
2.1	31/05/2017	WE	1.1	packet acceptance criteria: payload >= 64 bytes
2.2	25/09/2018	WE	1.1	additional NACK types for FDR Josh core imlementation