OpenCPI RDMA Protocol Specification

Authors:
John Miller, Mercury Federal Systems, Inc. (jmiller@jlc.net)
Jim Kulp, Mercury Federal Systems, Inc. (jkulp@mercfed.com)

Revision History

Revision	Description of Change	By	Date
1.01	Creation	jmiller	2007-11-28
1.02	Progressive Update based on Jim Kulp Comments	jmilller	2007-12-11
1.03	Sync up with OpenCPI doc formatting	jkulp	2010-07-24

Table of Contents

1 References

This document depends on several others. Primarily, it depends on the "OpenCPI Generic Authoring Model Reference Manual", which describes concepts and definitions common to all OpenCPI authoring models, and the "OpenCPI HDL Authoring Model Reference" which describes how HDL workers must be written.

Title	Published By	Link
OpenCPI Authoring Model Reference	OpenCPI	Public URL: http://www.opencpi.org/doc/amr.pdf
OpenCPI HDL Authoring Model Reference	OpenCPI	Public URL: http://www.opencpi.org/doc/hdlamr.pdf

2 Purpose

This document is a functional specification for the OpenCPI data plane RDMA data transfer protocol (OCPIRDT). The “data plane” is the virtual collection of the data communication paths between the various platform and infrastructure components within the system.

This document defines a protocol with several options and modes that allow for data transfer optimization based upon the required transfer complexity. There are three classes of RDMA transfer protocol modes: fixed length, variable length and data distribution and partitioning (DD&P) mode. Several optional protocol features are also specified.

The OCPIRDT specification is intended to provide data transfer interoperability between heterogeneous components executing within FPGAs, DSPs, and GPPs that all have access to each other’s addressable space.

This document uses the terms “producer port” to refer to a worker port that is sending data messages and a “consumer port” to refer to a worker port that is receiving data messages. Ports can be scaled, meaning that a port is actually a multiplicity of ports acting in concert, as a group that produces or consumes a sequence of messages. The term “port set” is used when scaled ports are explicitly referred to. Otherwise we simply use “port” to indicate where messages come from or go to.

This specification is fabric independent but assumes that both the producer and consumer have the ability to write (not read) each other’s address space (via some mechanism).

3 Scope

This specification describes the wire level protocol used by the OpenCPI RDMA data transfer layer. The protocol dependencies, behavior and requirements are documented. The setup and configuration data is also described at an abstract level but the connectivity establishment details are not specified at this level.

4 Requirements

The OCPIRDT external behavior is specified to meet these requirements:

- Multiple transfer modes shall be defined to support optimizations based on a connections transfer requirements.
- A producer port and a consumer port within a connection shall be allowed to use different transfer modes.
- Support heterogeneous transfer capabilities between FPGAs and GPPs. Note that DSPs are considered a subset of GPPs.
- All supported protocol modes shall provide flow control. This means that a consumer shall signal a producer when it has consumed all the data in a buffer.
- Shall support N-to-N multi buffering for all operational modes (same number of buffers on both sides).
- Shall optionally support M to N multi buffering for all operational modes (different number of buffers on both sides).
- Shall provide sequential multi buffer consumer access. This means that consumer multi buffers will be treated as a circular buffer set. The producer will therefore fill consumer buffers in a circular sequential fashion.
- Shall define an optimized mode of data transfer protocol to support a fixed length buffer transfer.
- Shall define a mode of data transfer protocol to support a variable length buffer transfer.
- Shall define a mode or modes of data transfer protocol to support scalability (port sets).
- All modes shall support data transfer capability for 1 producer port (set) and multiple consumer ports (sets). Multiple producer ports (sets) will not be supported. This does not mean that a producer port cannot be scaled, this requirement means that “fan in” from different port sets will not be supported.
- Shall be scalable to an arbitrary number of ports within a set.
- Shall not require software agents to use remote polling to monitor buffer state.
- Endpoint endian support shall not be considered at this level.
- The protocol shall not require error reporting at this level

5 Assumptions

The behavioral specification makes these assumptions on the environments in which OCPIRDT will be used/deployed.

- DMA engines may not support transfers less than 8 bytes.
- 64 bit transfers will arrive atomically and be seen by the software/processing running on the receiving end of the transfer atomically. I.e. a processor reading a 64 bit value that it has been written to its local memory will never see a partially updated 64 bit value.

6 Goals

- Throughput optimization is critical.
- External information needed to connect ports will be minimal.

7 Terms and Definitions

Term	Definition
Component	An independently defined unit of functionality
OCPIRDT	OpenCPI RDMA Data Transfer
DD&P	Data Distribution and Partitioning
FPGA	Field Programmable Gate Array
GPP	General Purpose Processor (e.g. Pentium, PowerPC)
IP	Intellectual Property (as captured in FPGA modules)
Worker	A functional instantiation (of an implementation) of a component

8 Overview

The OCPIRDT will provide a common set of protocol modes that will allow efficient interoperability between components within the system. The protocol consists of a minimum of three transfer types: data, buffer full flag and buffer empty flag. Figure 1 shows the minimal data transfer control flow.

Figure 1 Basic Transfer Signaling

The basic flow consists of a data buffer transfer from the producer to the consumer's buffer (local to the consumer). The next transfer is a buffer full flag sent from the producer to the consumer. Once the consumer has finished with the buffer the consumer transfers a buffer empty flag back to the producer. "Transfers" in this case are remote "RDMA writes" across the fabric, to a set of consecutive addresses.

There are several more complex transfer modes described later in the document. Transfer optimization is the reason to separate the modes instead of defining a single mode that is capable of supporting all possible scenarios.

It is important to note that modes 1 thru 3 do not provide any message fragmentation or reassembly. This means that a message must be moved in a single buffer transfer when these modes are used.

The infrastructure's setup and configuration software is expected to select the most suitable protocol based upon the connection requirements and the connected ports' capabilities.

There are several optional features in the OpenCPI RDMA protocol that are also described in this document such as M to N multi-buffering and fan out. Also, in some modes, metadata information such as the length of a variable length message is also transferred as well as the actual message data. Figure 2 describes a worst case scenario that has fan out and multi-buffering where there are X producer buffers, Y consumer buffers for port 1 and Z consumer buffers for port 2. In addition this figure demonstrates a case where the buffer full flag is separate from the metadata. (In some optimizations, metadata can actually serve as the buffer full flag) This figure also demonstrates the transfer requirements of the producer and consumer ports when metadata is separate from the buffer full flag.

The producer port needs to know the following address information for the buffers that a consumer has:

- Data buffer address(s)
- Metadata buffer address(s)
- Buffer full flag address(s)

The Consumer requires the following descriptor information about the producer for the buffers that the consumer has.

- Buffer empty flag address(s)

Note that in the diagram is appears that the data buffers, metadata and buffer full flags resides in contiguous memory. This is not necessarily the case since all three memory base addresses are passed to a producer.

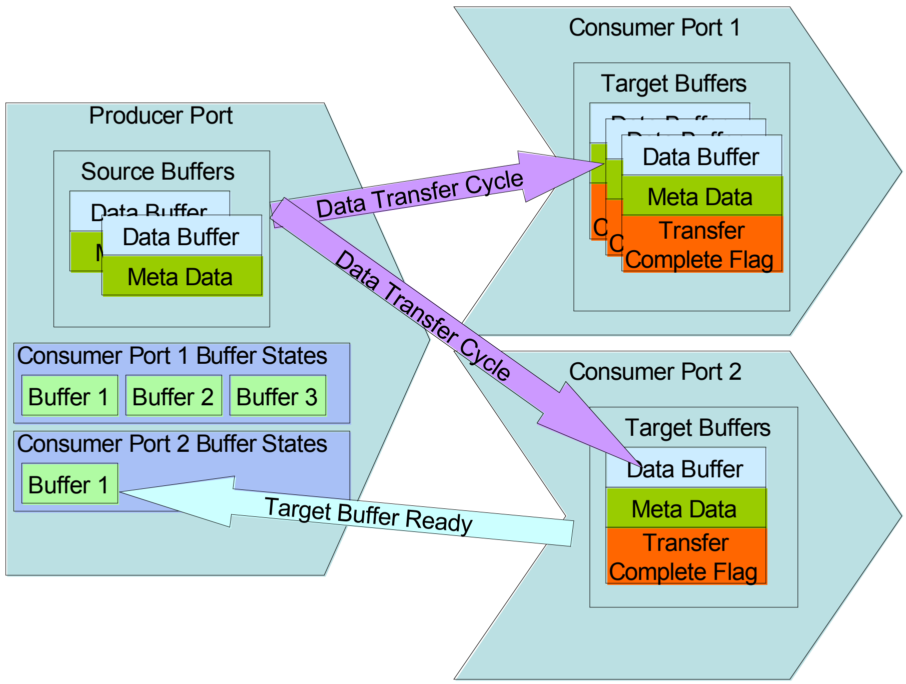

Figure 2 Data Transfer Example

9 Metadata Format

Metadata is an optional feature that is used when information such as the length of variable length messages are being transferred or more than 1 op-code (message type) is needed. The metadata that is sent from the producer to the consumer contains out of band data that includes parameters relating to the associated data buffer. There are three formats that are used, two of them (8.2 and 8.3) are for Data Distribution and Partitioning (DD&P) transfers and the third (8.1) is used for all other transfers.

9.1 Producer to Consumer Metadata format (Non-DD&P)

This metadata is sent from the producer to the consumer for each buffer that is transferred. It contains information regarding the length of the message and the Op-code. Note that the MSB must always be set to 1. This is required by the GPP when the Op-code=0 and the transfer length=0 (metadata only transfer). The Port Id field is optionally required by the consumer to identify the consumer port that this metadata is associated with. When mode 2 is used and the buffer full flag is written to a FIFO device that is servicing messages for multiple ports, this field is used to route the message. The value that the producer writes to this field is defined in the consumer descriptor field name “FullFlagValue”.

Word	Bits	64 bit words
0	0 - 31	Transfer length in bytes.
0	32- 39	Op-code
0	40-47	Port Id - Used to identify the port associated with this metadata. This is an optional field used in mode 2 to identify the port when the data full flag location is a FIFO.
0	48 - 62	Reserved
0	63	Always set to 1

9.2 Producer to Consumer Metadata format for DD&P Transfers

This metadata is sent from producer to consumer and is only used for scaled port transfer modes.

Word	Bits	64 bit words	Description
0	0 – 31	Transfer length	Number of bytes transferred.
0	32 – 39	Op-code	Op-code
0	40	End of Stream Marker	Stream framing flag.
0	41	End of Message Marker	Message data framing flag.
0	42 – 50	Part temporal id	Temporal id of this part of the buffer as it relates to the message source.
0	51 – 59	Producer rank	When a producer port is scaled this value indicates which producer port in the port set produced this data.
0	60 - 61	Reserved	
1	0 -7	Number of parts per message	Total number of parts the message was broken into.
1	8 - 15	Part rank	The part of the message this buffer represents.

9.3 Producer to Producer Metadata format for DD&P Transfers

This metadata is sent from producer to producer and is only used in some scaled transfer modes. The transfer token in used as barrier sync when a scaled producer port is configured for round robin distribution.

Word	Bits	64 bit words
0	0	Transfer Token

10 OpenCPI RDMA Data Transfer Protocol Modes

There are several protocol modes defined for OCPIRDT to support data transfer optimization. The infrastructure's setup and configuration entity is expected to select the most suitable protocol based upon the connection requirements and the connected ports' capabilities or static decisions. The modes that are defined below have been defined based upon a use case analysis of OpenCPI data transfer requirements applied to known limitations and constraints of existing DMA controllers. Table 1 highlights some of the more important attributes of each mode.

	Producer TXs per buffer	Consumer TXs per buffer	Supported op-codes	Variable length transfers?	Requires separate full flag	Meta data required
Mode 1	2	1	1	no	yes	no
Mode 2	2	1	256	yes	no	yes
Mode 3	3	1	256	yes	yes	yes
Mode 4	3+	1+	256	yes	yes	yes

Table 1 Protocol modes

10.1 Optional protocol feature set

Table 2 describes the optional features that each of the modes in table 1 may support. This table also indicates which features will be supported in the phase 1 implementation time frame.

	Mode 1	Mode 2	Mode 3	Mode 4	Implemented in phase 1
N to N multi-buffers	Yes	Yes	Yes	Yes	Yes
M to N multi-buffers	Yes	Yes	Yes	Yes	No
Variable length transfers	No	Yes	Yes	Yes	Yes
Number of op-codes>1	No	Yes	Yes	Yes	Yes
Fan out	Yes	Yes	Yes	Yes	No
Producer scalability	No	No	No	Yes	No
Consumer scalability	No	No	No	Yes	No

Table 2 Optional protocol features

10.2 Constraints imposed on memory addresses

This protocol requires fixed pitch addressing for consumer's data buffers, metadata buffers, buffer full flags and producer's buffer empty flags. Figure 3 is a diagram that shows the memory layout for the protocol. A description for each of the memory segments and the associated parameters can be found in the following sections. Note that the metadata segment is not used in all modes.

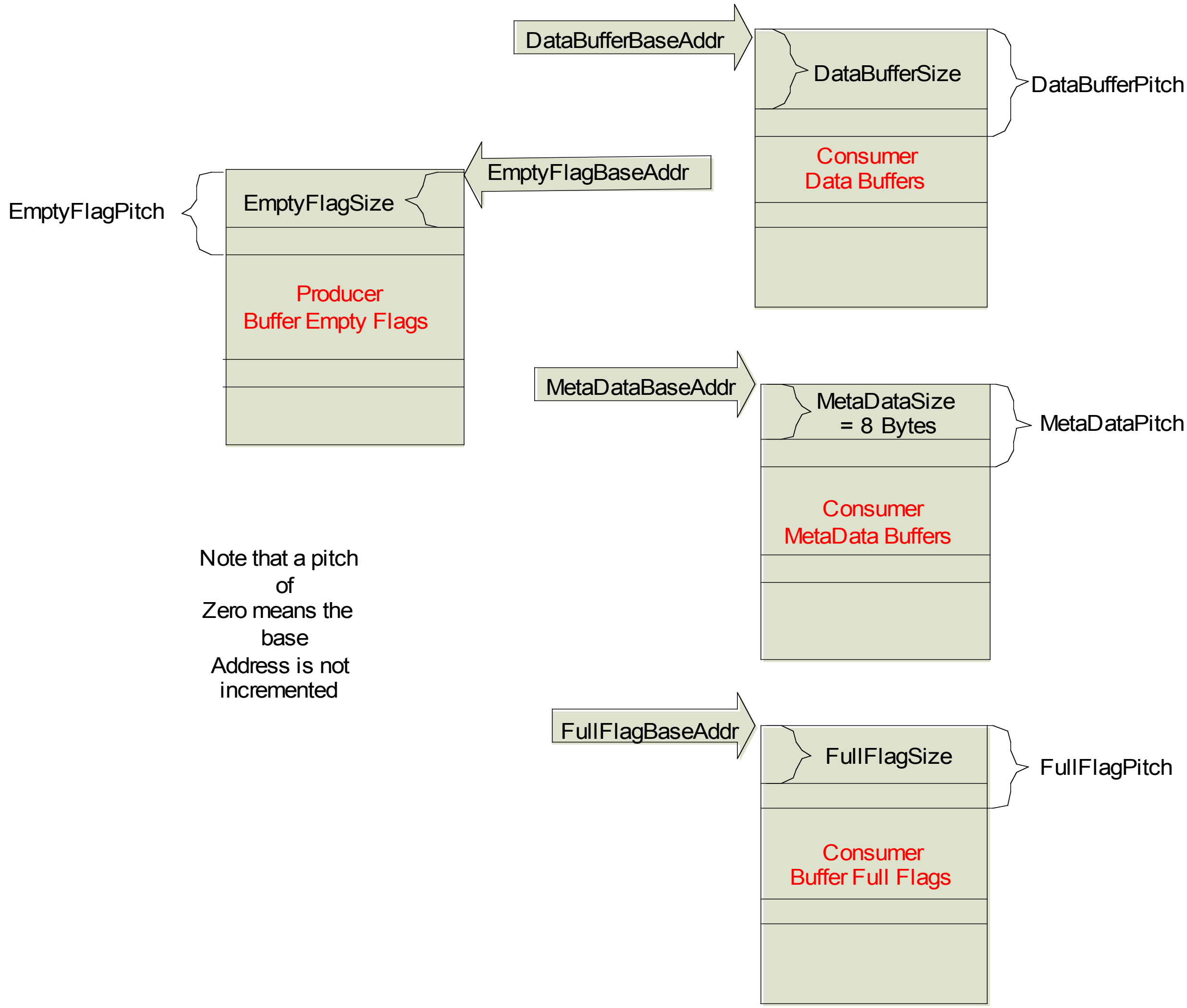

Figure 3 Memory Layout

10.3 Mode 1: Fixed length single OP-code

This is the OCPIRDT base protocol. Mode 1 consists of a fixed length data transfer from the producer to the consumer. The buffer full flag transfer contains no metadata information; it is only used to “hit” an address space to tell the consumer that a consumer buffer is full. The flag is always non-zero for consumers that need to see its contents as different from zero. There are other constraints on this flag word depending on the hardware being “hit”, e.g. 16 bit transfers for RIO doorbells.

10.3.1 Mode selection criteria

This mode can only be used if a fixed size data transfer is acceptable and there is only 1 op-code defined for the connection. This mode is generally selected when performance is critical on a connection in which the consumer port requires the buffer full flag separate from metadata.

10.3.2 Sequence diagram

Figure 4 depicts the transaction sequence for mode 1. In this diagram, the Transfer Manager is an abstract interface that a worker uses for acquiring, sending and receiving buffers. When a producer port produces a message it informs the PTransfer Manager. The PTransfer Manager transfers the message data and buffer full flag to the consumer. The CTransfer Manager makes the buffer available to the consumer port. Once the worker at that port has consumed the contents of the buffer it informs the CTransfer Manager which writes the buffer empty flag back to the producer.

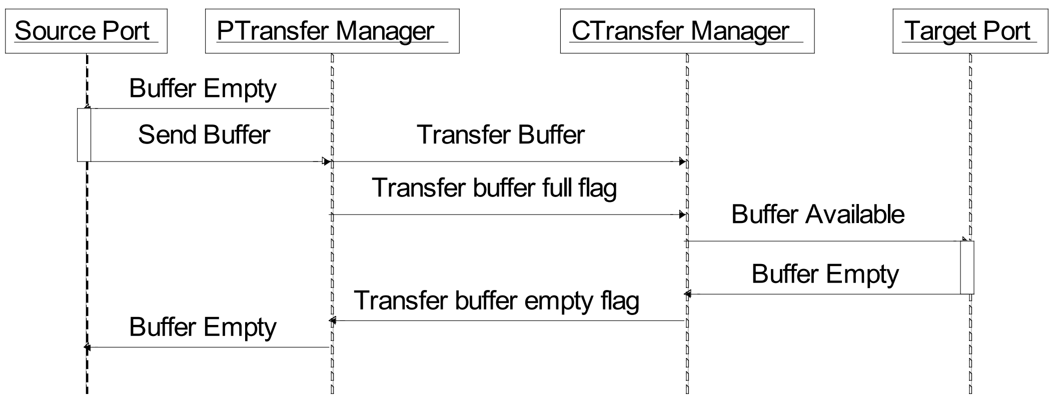

Figure 4 Mode 1 sequence diagram

10.3.3 Setup Information

10.3.3.1 Producer Port descriptor requirements

10.3.3.1.1 Consumer descriptor information needed by the producer

During the setup and configuration phase, the producer port infrastructure must know the following consumer descriptor information.

Note that the producer's buffer address(s) are not part of the descriptor parameters since this is an implementation detail, the producer port could be a FIFO.

Name	Count	Bits	Description
DataBufferBaseAddr	1		Consumer data buffer base address
DataBufferPitch	1		Consumer data buffer address pitch.
DataBufferSize	1		Maximum data transfer length in bytes
FullFlagBaseAddr	1		Consumer data buffer full flag base address
FullFlagPitch	1		Full flag address pitch.
FullFlagSize	1		Full flag size in bytes
FullFlagValue	1	8	The value that must be written to the full flag location. Typically used to identify the port.
NBuffers	1		Number of consumer buffers

10.3.3.2 Consumer Port requirements

10.3.3.2.1 Producer descriptor Information needed at the wire

During the setup and configuration phase, the consumer port configuration infrastructure must know the following producer descriptor information.

Name	Count	Size	Description
EmptyFlagBaseAddr	1		Producer target buffer empty flag base address
EmptyFlagSize	1		Number of bytes to be written to empty flag location
EmptyFlagPitch	1		Empty flag address pitch
EmptyFlagValue	1	8 bits	The value that needs to be written to the empty flag address. Typically used to identify the port.

10.4 Mode 2: Variable length (Metadata = Buffer full flag)

This mode is identical to mode 1 with two exceptions. First, the producer sends the actual number of bytes that get transferred to the consumer's buffers, and second the producer formats and writes the 64-bit metadata word as the buffer full flag described in section 8.1. This means that the consumer must be capable of using the metadata also as the buffer full indication.

10.4.1 Mode selection criteria

This mode is selected to meet the following requirements.

- DD&P is not required.
- Variable length transfers are required, ***or*** more than 1 op-code is needed, thus metadata is required,
- Consumer is capable of using a metadata transfer (arrival of metadata) as the buffer full indication.

10.4.2 Setup Information

10.4.2.1 Producer Port descriptor requirements

10.4.2.1.1 Consumer descriptor information needed at the wire

This is the same as mode 1. However since the metadata is written to the buffer full flag, the full flag size must be 64 bits.

10.4.2.2 Consumer Port requirements

10.4.2.2.1 Producer descriptor Information needed at the wire

Same as mode 1.

10.4.3 Sequence diagram

Figure 5 depicts the transaction sequence for mode 2. This diagram is similar to the sequence diagram for mode 1 with the following exceptions.

The PTransfer Manager must provide the actual length of the data transfer where length <= MaxDtLength if variable length transfers are configured. It must also provide the 8 Byte metadata that will be written to the metadata buffer location. Note also that for this mode the producer port must indicate the actual number of bytes that should be moved if the optional variable length transfers feature is enabled. In fact, the implementation may (inefficiently) send more bytes that indicated by the producer port as long as the length is inserted into the metadata.

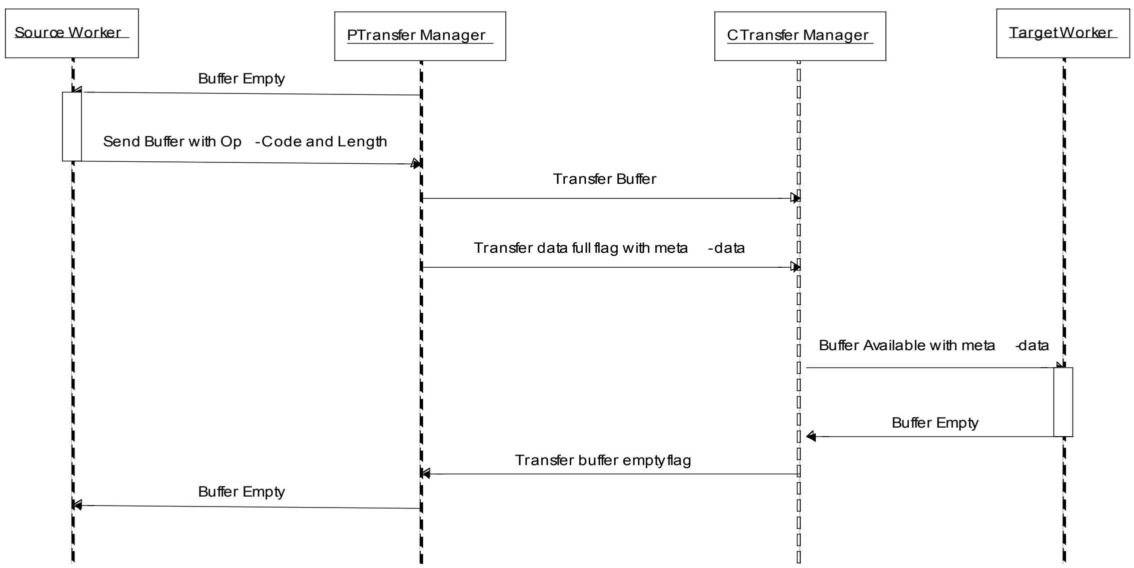

Figure 5 Mode 2 sequence diagram

10.5 Mode 3: Variable length (Metadata != Buffer full flag)

This mode is identical to mode 2 with one exception; the buffer full flag is transferred separately from the metadata. This mode is generally used when a consumer endpoint requires a buffer full flag (doorbell) location separate from the metadata. Therefore this mode creates a separate transfer to write a buffer full flag to a target location for each buffer that gets transferred. This mode has an additional data transfer for each buffer and therefore carries more overhead than mode 2.

10.5.1 Mode selection criteria

This mode is generally selected to meet the following requirements.

- DD&P is not required.
- Variable length transfers are needed, ***or*** more than 1 op-code is needed.
- Consumer requires a buffer full flag in addition to the metadata

10.5.2 Setup Information

10.5.2.1 Producer Port descriptor requirements

10.5.2.1.1 Consumer descriptor information needed by the producer

During the setup and configuration phase, the producer infrastructure must know the following consumer descriptor information.

Name	Count	Size	Description
DataBufferBaseAddr	1		Consumer data buffer base address
DataBufferPitch	1		Consumer data buffer address pitch
DataBufferSize	1		Maximum data transfer length in bytes
MetaDataBaseAddr	1		Consumer metadata base address
MetaDataPitch	1		Consumer metadata address pitch
NBuffers	1		Number of consumer data buffers
FullFlagBaseAddr	1		Consumer data buffer full flag base address
FullFlagSize	1		Full flag size in bytes
FullFlagPitch	1		Full flag address pitch
FullFlagValue	1	8 bits	The value that must be written to the full flag location. Typically used to identify the port.

10.5.2.2 Consumer Port requirements

10.5.2.2.1 Producer descriptor Information needed by the consumer

During the setup and configuration phase, the consumer port configuration infrastructure must know the following producer descriptor information.

Name	Count	Size	Description
EmptyFlagBaseAddr	1		Producer target buffer empty flag base address
EmptyFlagSize	1		Size of empty flag in bytes
EmptyFlagPitch	1		Empty flag address pitch
EmptyFlagValue	1	8 bits	The value that must be written to the empty flag location Typically used to identify the port.

10.5.3 Sequence diagram

Figure 6 depicts the transaction sequence for mode 3. This sequence is similar to mode 2 except that the metadata is transferred separately. Since the metadata is transferred separate from the buffer full flag, additional consumer metadata buffer addresses are required for this mode.

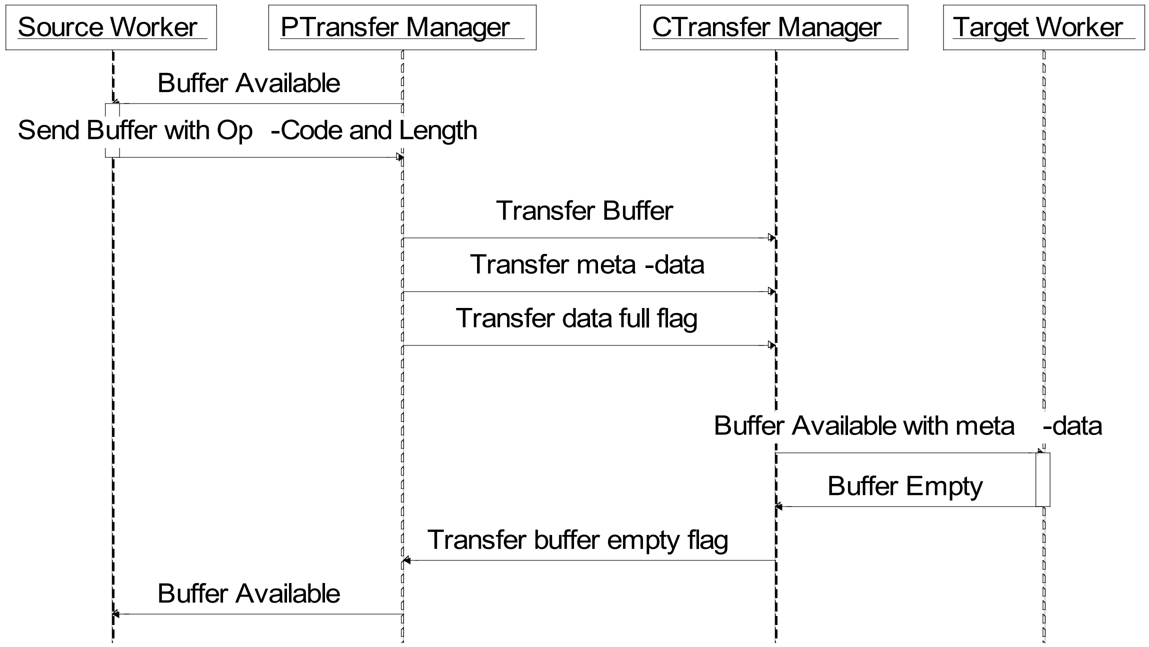

Figure 6 Mode 3 Transaction sequence diagram

10.6 Mode 4: DD&P port data transfer

DD&P data transfer mode is used for data distribution and data partitioning for both producer and consumer ports. This mode is used when a CPI infrastructure supports DD&P. This means that a worker can be duplicated N times on collocated or non-collocated nodes within the system, or when multiple consumer ports are connected to the producer port. This mode supports a scatter/gather pattern for partitioned messages (messages produced or consumed in parts).

Future versions of this document will provide more coverage of the various DD&P patterns — they are not in the current requirement set.

This mode supports data distribution as well as data partitioning. In general, a producer port set produces a temporal stream of whole messages, which is consumed by one or more consumer port sets.

10.6.1 Data Distribution Model

The data distribution model describes the pattern of where the messages come from and where they go to. The three data distribution patterns are: broadcast (each message goes to all consumers), round-robin (messages are distributed to consumers in a rotating pattern), and least busy (messages are distributed to consumers that have the fewest messages already received but not yet processed). These patterns can be applied at two levels: in distributing messages across multiple consumer port sets, and in distributing messages across the members within a port set.

Thus a connection specifies the pattern within the producer port set, the pattern across the consumer port sets, and the pattern within each consumer port set.

10.6.2 Data partitioning Model

The data partitioning model deals with messages that are in fact multidimensional Cartesian data objects, and describes how the whole messages are partitioned (divided into parts) during communication. A whole message may be partitioned at a producer port set. This means that different members of the port set produce different parts of the whole message so that the output of the set is a stream of parts of messages. If a connection specifies partitioning, then the data distribution pattern may specify how the parts are distributed to a consumer port set rather than how the whole messages are distributed. Messages produced in parts are usually produced by dividing up the message (whole Cartesian data object) along the most major dimension: i.e. if a message is partitioned across a producer port set of 4 members, then each member produces one quarter of each message. When these parts are produced they are conveyed according to some data distribution pattern to the consumer port sets, and then according to some data distribution and partitioning to some consumer port set.

Messages are not partitioned across multiple consumer port sets, only within a consumer (or producer) port set.

A complex example would be a producer port set of three members producing parts (1/3) of messages that were consumed by two port sets according to the broad cast pattern (both consumer port sets get each whole message), where one consumer port set of 4 members would receive the data partitioned across the members (each receiving parts that were ¼ of each whole), while the second consumer port set of 2 members received the messages round robin but not partitioned, with each member receiving every other whole message.

10.6.3 Basic gather signaling

Gather is the term used when “whole messages” are produced in parts from the members of the producer port set, and must be “gathered up” or assembled into whole messages for consumption. This scenario is used when a producer is scaled. The fundamental operation of this mode requires the producer’s setup and configuration infrastructure to know to what part of a consumers buffer (where the messages will be assembled into a whole) the data gets transferred to. This primarily affects the complexity of the transfer setup. However, since a consumer does not consider a buffer full until all producers transfer a buffer full flag, the signaling becomes more complicated.

In general, if the consumer is a FPGA device, the buffer full flag has the same address location for all producers. This is accomplished through the use of a hardware counter and once the counter reaches a count equal to the number of producers the buffer becomes available to the target worker.

When the consumer is a GPP and the consumer’s buffer full flag is resident in memory, there are dedicated flag locations for all producers in a producer port set. This is required to avoid race conditions since in this case there are no dedicated registers with hardware arbitration. So each of the producers transfer their buffer full flag to a unique location and the GPP’s Consumer Transfer Manager is responsible for making sure that all of the message parts have been transferred before making the message available to the consumer port. If a GPP has some mechanism to operate like the FPGA they are free to use it.

Flow control signaling is also affected in this scenario. Once a consumer is finished with a buffer, it is required to tell all producers that the buffer is empty.

10.6.3.1 FPGA gather signaling

Figure 7 illustrates the typical protocol for a DD&P producer gather configuration when the consumer is an FPGA. Each of the producers in a port set transfers their portion of the data buffer to the appropriate location on the consumer (red arrows). Once their part is transferred each producer writes the buffer full flag, indicated by the green arrows, to the same address location causing the buffer full register to increment its count. Once the count reaches the known number of producers in the port set, the Buffer Manager makes the message available to the worker. Note that blue arrows in the diagram show that in this mode, a consumer must send a buffer empty flag to every producer that is supplying parts.

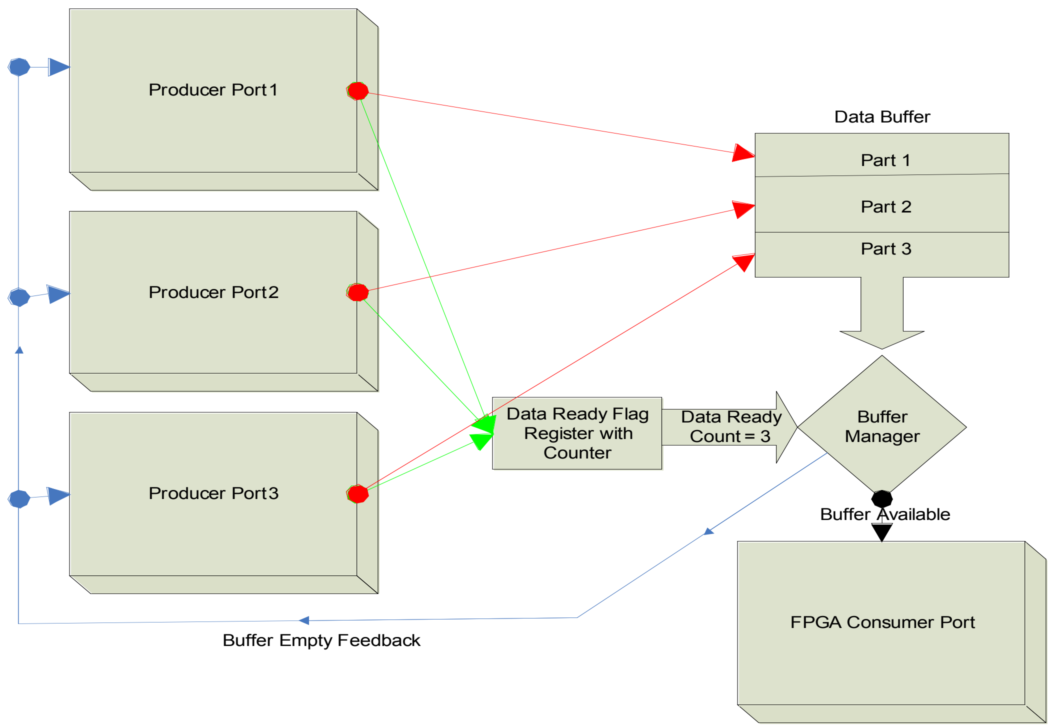

Figure 7 FPGA Gather Signaling

10.6.3.2 GPP Gather signaling

Figure 8 illustrates the typical protocol for a DD&P producer gather configuration when the consumer is a GPP. Each of the producers in the port set transfer their portion of the data message to the appropriate location on the target (red arrows). Once their part is transferred each producer writes the buffer full flag, indicated by the green arrows, to unique address locations. It is the responsibility of the GPP Buffer Manager to ensure that all the consumer buffer empty flags are written before the buffer is reused. Note that blue arrows in the diagram show that in this mode, a consumer must send a buffer empty flag to every producer that is supplying parts.

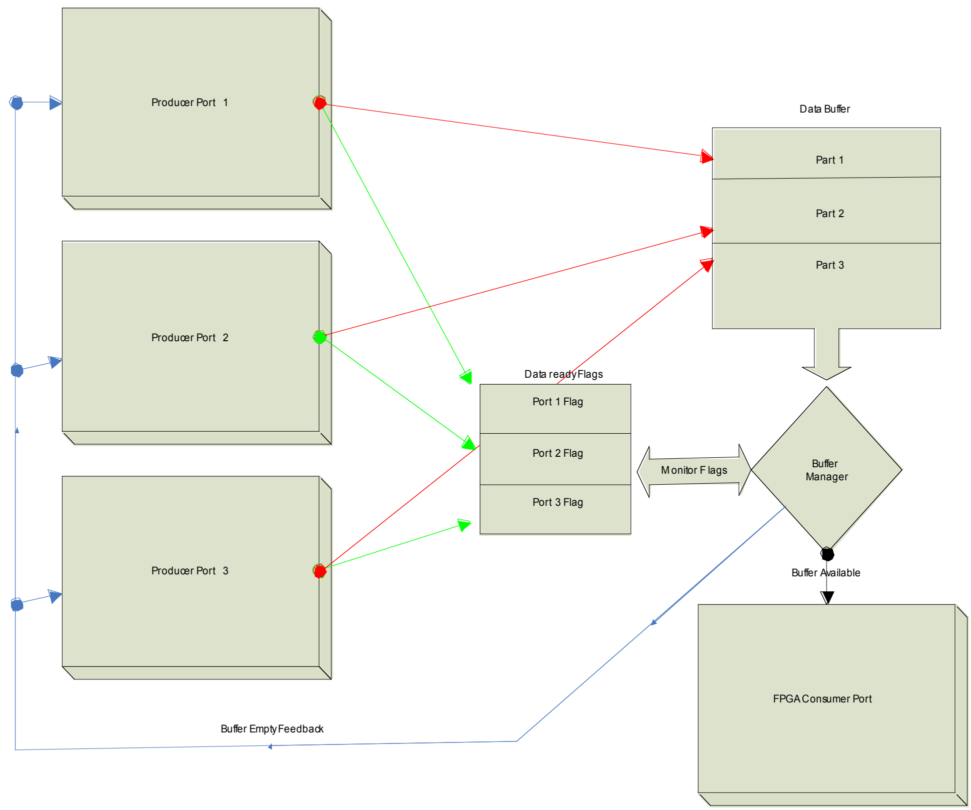

Figure 8 GPP Gather signaling

10.6.4 Basic scatter signaling

Scatter is the term that is used to describe the partitioning and distribution of message data produced as whole messages from one or more producers and distributed as parts (partial messages) to 2 or more consumers. This scenario is used when a consumer is scaled.

The fundamental operation of this mode requires the producer's setup and configuration infrastructure to know what parts of the whole messages it produces to transfer to what consumer's buffer . This primarily affects the complexity of transfer setup. Once the parts of the data are transferred to the consumers, each consumer must provide flow control to the producer once its buffer is empty.

In general, if the producer is a FPGA device, the buffer empty flag has the same address location for all consumers (zero pitch). This is accomplished through the use of a hardware counter and once the counter reaches a count equal to the number of consumers in the port set, the consumer buffers become available to the producer.

When the producer is a GPP and the producer's buffer empty flag is resident in memory, and there are dedicated flag locations for all consumers in the port set. This is required to avoid race conditions since in this case there are no dedicated registers with hardware arbitration. So each of the consumers transfers their buffer empty flag to a unique location and the GPP's Producer Transfer Manager is responsible for making sure that all of the consumers have provided flow control before a consumer buffer part is reused. If a GPP has some mechanism to operate like the FPGA they are free to use it. A GPP that has separate buffer empty memory locations is free to transfer parts to those consumers as they become empty, or wait for all consumers to indicate buffer empty and perform a single transfer to all consumer buffers.

10.6.4.1 FPGA Scatter signaling

Figure 9 illustrates a simple scatter pattern and the relevant signaling involved. The red arrows show how a producer sends the parts to different consumers. The red arrows imply both data and buffer full transfers. The blue arrows demonstrate how all consumers must provide flow control with a buffer empty flag transfer. In this case, the producer is an FPGA and the buffer empty flag is a register with a counter. All consumers write to the same address location and once the counter is equal to the number of scaled consumers the FPGA Producer Transfer manager can use those consumer buffers for future transfers.

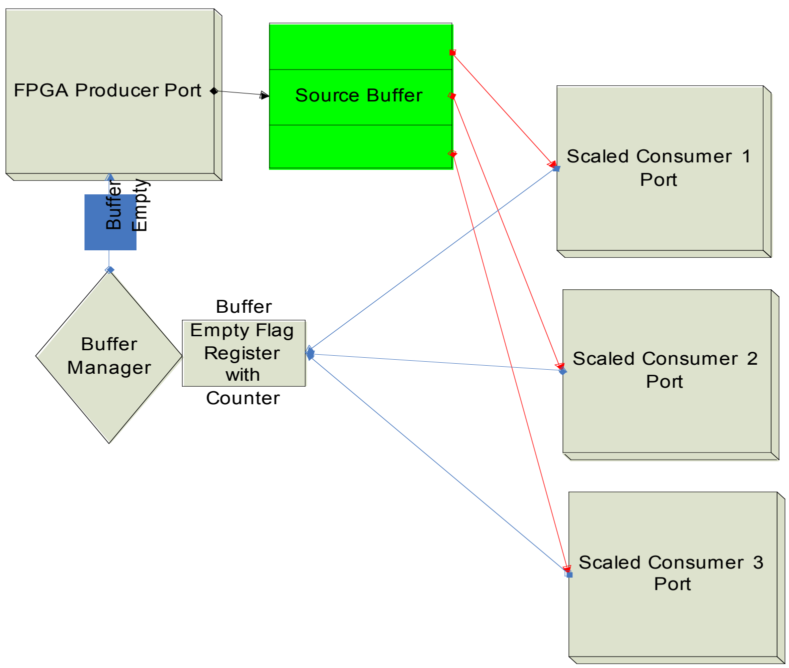

Figure 9 FPGA Scatter signaling

10.6.4.2 GPP Scatter signaling

Figure 10 illustrates a simple scatter pattern and the relevant signaling involved. The red arrows show how a producer sends the parts to different consumers. The red arrows imply both data and buffer full transfers. The blue arrows demonstrate how all consumers must provide flow control with a buffer empty flag transfer. In this case, the producer is a GPP and each of the buffer empty flags are unique memory locations. All consumers write to different address locations and once a transfer has been completed to each of the locations; the GPP Producer Transfer Manager can use those consumer buffers for future transfers.

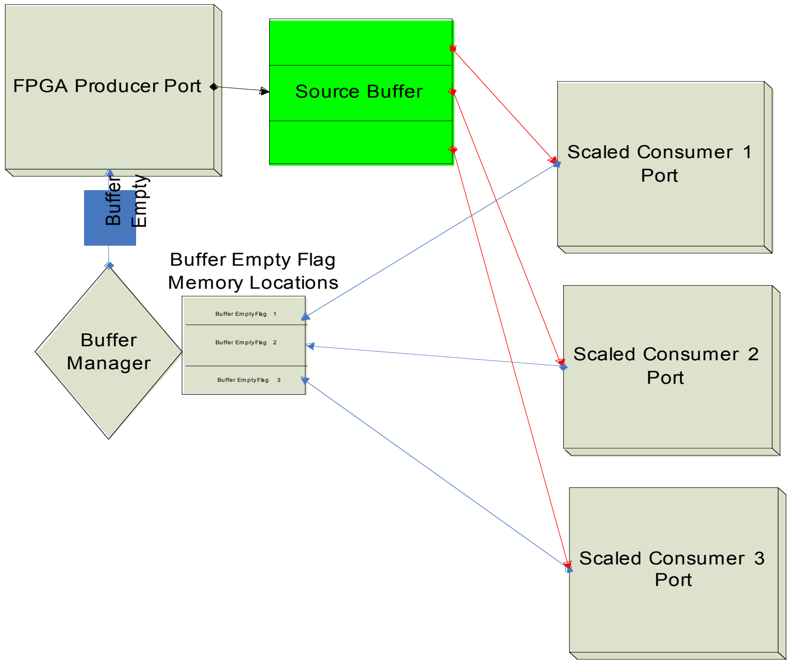

Figure 10 GPP Scatter signaling

10.6.5 Mode selection criteria

This mode is selected to meet the following requirements.

- Producer or Consumer DD&P is required.
- Message partitioning is required (Message Length > Buffer Length)

10.6.6 Setup Information

10.6.6.1 Producer Port descriptor requirements

10.6.6.1.1 Consumer descriptor information needed by the producer

During the setup and configuration phase, the infrastructures producer port configuration infrastructure must know the following consumer descriptor information.

Name	Count	Size	Description
NProducers	1		Number of producers that are going to supply data parts to consumers
NConsumers	1		Number of consumers that are going to be receiving parts of the producers data
DataBufferBaseAddr	NConsumers		Consumer data buffer base address(s)
DataBufferPitch	1		Data buffer address pitch
DataBufferSize	1		Maximum data transfer length in bytes
MetaDataBaseAddr	NConsumers		Consumer metadata base address(s)
MetaDataPitch	1		Metadata address pitch
FullFlagBaseAddr	NConsumers		Consumer full flag base address(s)
FullFlagSize	1		Size of full flag in bytes
FullFlagPitch	1		Full flag address pitch
FullFlagValue	1		The value that must be written to the full flag location.
ProdBarrierSyncAddr	NProducers		(Optional) Some producer distribution patterns require this token to determine when to produce
ProdBarrierSyncPitch	1		Barrier sync address pitch

10.6.6.2 Consumer Port requirements

10.6.6.2.1 Producer descriptor Information needed by the consumer

During the setup and configuration phase, the infrastructures consumer port configuration infrastructure must know the following producer descriptor information.

Name	Count	Size	Description
EmptyFlagBaseAddr	1		Producer's empty flag base address
EmptyFlagSize	1		Size of empty flag in bytes
EmptyFlagPitch	1		Empty flag address pitch
EmptyFlagValue	1		The value that must be written to the empty flag location

11 Setup and Configuration

11.1 Mode Selection based on port constraints

The setup and configuration entity within the infrastructure must select the port's protocol modes based on the set of features required for a connection and the ports constraints. It is acceptable for different connections between two devices such as a FPGA and GPP to select different modes and features based on the connection requirements and the direction of data flow. For example, if a FPGA device requires a buffer full flag, then the producer port on the GPP must select mode 3 if either variable length transfers are required or more than 1 op-code is needed. However if the producer is on the FPGA, mode 2 can be selected since the GPP does not need the separate flag. Table 5 illustrates mode selection based on the desired feature set and the constraints introduced by the port endpoint. Note that all other optional features that may be supported such as multi-buffering can be supported by all modes.

	No metadata required: Fixed length transfers and Op-codes = 1	**Metadata required (Variable length Transfers OR Op-codes > 1)**	**DD&P required**
Buffer full flag is required	Mode 1	Mode 3	Mode 4
Buffer full flag not required	Mode 1	Mode 2	Mode 4

Table 5 Mode Selection

11.2 Port connection setup and configuration considerations

The CPI infrastructure is responsible for the setup and configuration of port connections. Table 5 illustrates how the modes that are selected based on some of the optional features described in table 2. All other optional features such as multi-buffering and fan out can be supported in all modes. Therefore a strategy must be implemented by the CPI infrastructure when connecting producer ports to consumer ports to ensure compatibility. For example, if a consumer port can support multi-buffering but the producer port cannot, the infrastructure must be capable of setting the number of buffers on the consumer to 1. This setup within the infrastructure is not only critical for proper setup but is essential to ensure the highest quality of service requested for the connection.

11.2.1 Port Capability Profile

The CPIRDT specification defines both required capabilities and optional features that a port may support. The following table defines the list of capabilities that an infrastructure needs to know about the producer and consumer ports to make a mode and feature selection.

Some of the port capabilities are implied if a port advertises support for a particular mode. For example, if a port indicates that it can support mode 2 as a consumer, it is implied that it is capable of reading the buffer full flag value.

Feature	Optional	Producer only = 1, Consumer only = 2, Both = 3	Description
Mode 1	At least 1 mode support is required.	3	Support for mode 1.
Mode 2	At least 1 mode support is required.	3	Support for mode 2.
Mode 3	At least 1 mode support is required.	3	Support for mode 3.
Mode 4	At least 1 mode support is required.	3	Support for mode 4.
Requires Full Flag value	true	2	Indicates if the consumer requires the Full flag to be a specified value.
Requires Empty Flag value	true	1	Indicates if the producer requires the Empty flag to be a specific value.
Variable length transfers	true	1	Variable length transfers supported
Can mask port id in metadata	true	1	Producer can mask the port id into the metadata word.
Can write full flag value	true	1	Producer is capable of writing a specific full flag value specified by the consumer.
Fan out	true	1	Producer is capable of supplying multiple consumers.
Supports NxN multi-buffering	true	1	Producer is capable of NxN multi-buffered transfers.
Supports MxN multi-buffering	true		Producer is capable of MxN multi-buffered transfers.