Abstract

This document defines a Remote Direct Memory Protocol (iWarp) to run on the Direct Data Placement Protocol (DDPP) [DDPP]. This initial draft is an incomplete sketch of iWarp to be used only as the basis of discussion of protocol and architectural issues with DDPP and RDMA.
1. Introduction

This document defines a Remote Direct Memory Protocol (iWarp) to run on the Direct Data Placement Protocol (DDPP) [DDPP]. This initial draft is an incomplete sketch of iWarp to be used only as the basis of discussion of protocol and architectural issues with DDPP and RDMA.

iWarp follows the architecture and terminology of ‘The Architecture of Direct Data Placement (DDP) And Remote Direct Memory Access (RDMA) On Internet Protocols’ (DRARCH) [DRARCH]. A thorough understanding of DRARCH is necessary to understand this document.

iWarp defines three data transfer operations:

- RDMA Write
- RDMA Read
- Send (an undecorated message)

2. Flow Control in iWarp

While it is straightforward for client protocols to implement flow control over iWarp protocol resources, iWarp defines its own flow control because many client protocols prefer not to handle this
iWarp flow control is credit-based, with two distinct pools of credits:

- Send and Notifying RDMA Write credits,
- RDMA Read Request credits.

iWarp MAY submit one complete client protocol Send or Notifying RDMA Write (an RDMA Write which requests a reception indication) to DDPP for each Send and Notifying RDMA Write credit. Initial Send and Notifying RDMA Write credits are established when iWarp is enabled and may be returned in any iWarp message. Client protocols MAY chose to use Send and Notifying Write flow control or not.

iWarp MAY submit one RDMA Read Request to DDPP for each RDMA Read Request credit. Initial RDMA Read Request credits are established when iWarp is enabled and one credit is returned by each completed RDMA Read. Client protocols MAY chose to use RDMA Read Request flow control or not.

3. Use of DDPP Message Identifiers In iWarp

iWarp uses the first 11 bits of DDPP’s Message Identifier for its own purposes. The remaining (at least 4, probably 20) bits remain for use by client protocols. iWarp’s 11 Message Identifier bits are:

```
0 1 2 3 4 5 6 7 8 9 10
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|R|F|N|   Credits   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

- **R** - Read Reply Flag : 1 bit (boolean flag)
  
  if set to 1, the message is RDMA Read data, otherwise, it is RDMA Write data.

- **F** - Final RDMA Data Flag : 1 bit (boolean flag)
  
  if set to 1, the message is the last in a group for a single RDMA Write or RDMA Read Response.

- **F** - Notifying RDMA Write Flag : 1 bit (boolean flag)
if set to 1, the message is the last in a group for a single, notifying RDMA Write.

Credits : 8 bits (unsigned integer)

Amount by which to increase the Send and Notifying RDMA Write credits. MUST be 0 if Send and Notifying RDMA Write flow control is disabled.

4. RDMA Write In iWarp

An iWarp RDMA Write operation is a group of one more DDP-decorated messages with the Message Identifier field set as defined above.

A DDP-decorated message that is part of an RDMA Write MUST have Notify set when:

- it is the final DDP-decorated message in an RDMA Write which is requesting a completion indication, or
- the Credits value is not zero.

5. RDMA Read In iWarp

An iWarp RDMA Read operation is:

- an RDMA Read Request containing source and destination buffer addresses and RDMA Read size,
- an RDMA Read Response of one or more DDP-decorated messages targeting the destination buffer address with the data from the source buffer address.

An RDMA Read Request is an undecorated message:
Tp - Message Type : 4 bit (unsigned integer)

Undecorated message type. Must be 0x01 for an RDMA Read Request.

R - Reserved

Sender SHOULD set to 0, receiver MUST ignore.

Credits : 8 bit (unsigned integer)

Amount by which to increase the Send and Notifying RDMA Write credits.

Source STag : 32 bit (unsigned integer)

The steering tag identifying the source buffer from which to retrieve the RDMA Read data.

Source Offset : 64 bits (unsigned integer)

The offset in the source buffer from which to retrieve the RDMA Read data.

Read Size : 32 bits (unsigned integer)
The number of octets of data to be read from the source address.

Destination STag : 32 bit (unsigned integer)

The steering tag identifying the destination buffer in which to place the RDMA Read data.

Destination Offset: 64 bits (unsigned integer)

The offset in the destination buffer at which to place the RDMA Read data.

An RDMA Read Response is a group of one or more DDP-decorated messages with the Message Identifier field set as defined above. A DDP-decorated message that is part of an RDMA Read Response MUST have Notify set when:

- it is the final DDP-decorated message in an RDMA Read Response, or
- the Credits value is not zero.

The client protocol portion of the Message Identifier field of the DDP-decorated messages in an RDMA Read Response may be chosen by the client protocol. This allows the client protocol to distinguish among RDMA Read Responses for multiple outstanding RDMA Read Requests. Allowing the client protocol to select a portion of the Message Identifier permits a different interface from DRARCH’s synchronous rdma_read(). However, DRARCH’s rdma_read() can be implemented in iWarp by having each outstanding call to rdma_read() automatically select a different client protocol portion of the Message Identifier.

An RDMA Read Response MUST transfer exactly Read Size octets, or result in an error.

6. Send In iWarp

An iWarp Send is an undecorated message of up to 2^{31}-1 octets.

To permit efficient implementation, each Send is identified by a Send Sequence Number. The Send Sequence Number is not visible to client protocols. The first Send after iWarp is enabled MUST have Send Sequence Number 0. Each subsequent Send MUST have a Send Sequence number of 1 + the Send Sequence Number of the previous Send.
The data from a single client protocol-submitted Send is sent as a group of one or more Send messages where:

- Each Send message of the group MUST have the same Send Sequence Number.
- Each Send message of the group MUST have Send Offset equal to the offset in the client protocol-submitted Send of its first octet of data.
- Send messages other than the last of the group MUST NOT have the Final Message Flag set.
- The last Send message of the group MUST have the Final Message Flag set.

An individual Send message is an undecorated message:

```
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Tp=0x02|F|  R  |    Credits    |               R               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Send Sequence Number                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Send Offset                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                          Data Payload                         ~
~                                                               ~
|               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
|               |                                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Tp - Message Type : 4 bit (unsigned integer)

Undecorated message type. Must be 0x02 for a Send.

F - Final Message Flag : 1 bit (boolean flag)

if set to 1, this is the final Send message of a group carrying the data for a single client protocol-submitted Send

R - Reserved

Sender SHOULD set to 0, receiver MUST ignore.
Credits : 8 bit (unsigned integer)

Amount by which to increase the Send and Notifying RDMA Write credits.

Send Sequence Number : 32 bit (unsigned integer)

The sequence number of the client protocol-submitted Send.

Send Offset : 32 bit (unsigned integer)

The offset of Data Payload in the client protocol-submitted Send.

Data Payload : 0-2^31-1 octets (opaque data)

data from the client protocol-submitted Send.

7. Credit Return Message

If Send and Notifying Write credits can not be returned in a client protocol data transfer message, possibly because no client protocol data transfer is in progress, credits can be returned with a Credit Return Message.

A Credit Return Message is an undecorated message:

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Tp=0x03|   R   |    Credits    |               R               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Tp - Message Type : 4 bit (unsigned integer)

Undecorated message type. Must be 0x03 for a Credit Return Message.

R - Reserved

Sender SHOULD set to 0, receiver MUST ignore.

Credits : 8 bit (unsigned integer)

Amount by which to increase the Send and Notifying RDMA Write credits.
8. Errors In iWarp

[TODO]

9. Transport Characteristics In iWarp

The effect of transport characteristics on operation ordering in iWarp is discussed below.

In addition operation ordering, transport characteristics also interact with iWarp in other ways:

- RDMA Write, RDMA Read and Sends larger than a single transport message don’t work with unordered or unreliable transports.
- Flow control doesn’t work with unreliable transports.
- Flow control doesn’t work with multisource transports.
- RDMA Read flow control doesn’t work with multidestination transports.
- [TODO] Others?

10. Operation Ordering In iWarp

The ordering among:

- set()s,
- get()s,
- Sends,
- RDMA Write reception indications, and
- RDMA Read completion indications

and their relationship to corresponding operations on the sender is defined in iWarp according to underlying transport characteristics:

- reliable or unreliable, and
- ordered or unordered.

TODO: Now complicated stuff, especially about get() ordering.
10.1. Ordering On Reliable, Ordered Transports

On a reliable, ordered transport, iWarp:

- [TODO]

10.2. Ordering On Reliable, Unordered Transports

On a reliable, unordered transport, iWarp:

- [TODO]

10.3. Ordering On Unreliable, Ordered Transports

On an unreliable, ordered transport, DDPP:

- [TODO]

10.4. Ordering On Unreliable, Unordered Transports

On an unreliable, unordered transport, in general, no additional, transport-dependent rules apply to iWarp. [TODO?]

11. Transport Topology In iWarp

Transports support some combination of:

- single source, or multisource, and
- single destination, or multidestination (multicast or anycast).

When running iWarp on a multisource transport, flow control MUST NOT be enabled.

When running iWarp on a multidestination transport, RDMA Read flow control MUST NOT be enabled.

12. Negotiating iWarp

Negotiating the use of iWarp is the sole responsibility of the client protocol. iWarp is a duplex protocol, and must be enabled reciprocally in both directions by a pair of participants. Some client protocols (e.g. RDMA) MAY chose to require iWarp a priori, while others MAY define an in- or out-of-band negotiation process to dynamically enable iWarp. Whatever the case, a client protocol using iWarp MUST establish:
Use of Send and Notifying RDMA Write flow control,

Initial Send and Notifying RDMA Write credits (if enabled),

Use of RDMA Read Request flow control,

Initial RDMA Read Request credits (if enabled).

13. Security Considerations

[TODO]

14. IANA Considerations

[TODO]

15. References

[DDPP]
Bailey, S., "The Direct Data Placement Protocol (DDPP) Core",

[DRARCH]
Bailey, S., "The Architecture of Direct Data Placement (DDP)
And Remote Direct Memory Access (RDMA) On Internet Protocols",

Author’s Address

Stephen Bailey
Sandburst Corporation
600 Federal Street
Andover, MA  01810
USA

Email: steph@sandburst.com

Full Copyright Statement

Copyright (C) The Internet Society (2002). All Rights Reserved.

This document and translations of it may be copied and furnished to
others, and derivative works that comment on or otherwise explain