IP over InfiniBand(IPoIB) Architecture

Status of this memo

Copyright Notice

Abstract

        InfiniBand is a high speed, channel based interconnect between
        systems and devices.

        This document presents an overview of the InfiniBand
        architecture. It further describes the requirements and
        guidelines for the transmission of IP over InfiniBand.
        Discussions in this document are applicable to both IPv4 and
        IPv6 unless explicitly specified. The encapsulation of IP over

InfiniBand and the mechanism for IP address resolution on IB
fabrics are covered in other documents.

Table of Contents

1.0 Introduction to InfiniBand

The InfiniBand Trade Association(IBTA) was formed to develop
an I/O specification to deliver a channel based, switched
fabric technology. The InfiniBand standard is aimed at meeting
the requirements of scalability, reliability, availability and
performance of servers in data centers.

1.1 InfiniBand Architecture Specification

The InfiniBand Trade Association specification is available for download from http://www.infinibandta.org.

1.2 Overview of InfiniBand Architecture

For a more complete overview the reader is referred to chapter 3 of the InfiniBand specification.

InfiniBand Architecture (IBA) defines a System Area Network(SAN) for connecting multiple independent processor platforms, I/O platforms and I/O devices. The IBA SAN is a communications and management infrastructure supporting both I/O and inter-processor communications for one or more computer systems.

An IBA SAN consists of processor nodes and I/O units connected through an IBA fabric made up of cascaded switches and IB routers (connecting IB subnets). I/O units can range in complexity from single ASIC IBA attached devices such as a LAN adapter to a large memory rich RAID subsystem.

An IBA network may be subdivided into subnets interconnected by routers. These are IB routers and IB subnets and not IP routers or IP subnets. This document will refer to InfiniBand routers and subnets as 'IB routers' and 'IB subnets' respectively. The IP routers and IP subnets will be referred to as 'routers' and 'subnets' respectively.

Each IB node or switch may attach to a single or multiple switches or directly with each other. Each IB unit interfaces with the link by way of channel adapters (CAs). The architecture supports multiple CAs per unit with each CA providing one or more ports that connect to the fabric. Each CA appears as a node to the fabric.

The ports are the endpoints to which the data is sent. However, each of the ports may include multiple QPs (queue pairs) that may be directly addressed from a remote peer. From the point of view of data transfer the QP number (QPN) is part of the address.

IBA supports both connection oriented and datagram service between the ports. The peers are identified by QPN and the port identifier. There are a two exceptions. QPNs are not used when packets are multicast. QPNs are also not used in the raw datagram mode.

A port, in a data packet, is identified by a local ID (LID)

and optionally a Global ID (GID). The GID in the packet is needed only when communicating across an IB subnet though it may always be included.

The GID is 128 bits long and is formed by the concatenation of a 64 bit IB subnet prefix and a 64 bit EUI-64 compliant portion (GUID). The LID is a 16 bit value that is assigned when the port becomes active. Note that the GUID is the only persistent identifier of a port. However, it cannot be used as an address in a packet. If the prefix is modified then the GID may change. The subnet manager may attempt to keep the LID values constant across reboots but that is not a requirement.

The assignment of the GID and the LID is done by the subnet manager. Every IB subnet has at least one subnet manager component that controls the fabric. It assigns the LIDs and GIDs. The subnet manager also programs the switches so that they route packets between destinations. The subnet manager and a related component, the subnet administrator (SA) are the central repository of all information that is required to setup and bring up the fabric.

IB routers are components that route packets between IB subnets based on the GIDs. Thus within an IB subnet a packet may or may not include a GID but when going across an IB subnet the GID must be included. A LID is always needed in a packet since the destination within a subnet is determined by it.

A CA and a switch may have multiple ports. Each CA port is assigned its own LID or a range of LIDs. The ports of a switch are not addressable by LIDs/GIDs or in other words, are transparent to other end nodes. Each port has its own set of buffers. The buffering is channeled through virtual lanes(VL) where each VL has its own flow control. There may be up to 16 VLs.

VLs provide a mechanism for creating multiple virtual links within a single physical link. All ports must support VL15 which is reserved exclusively for subnet management datagrams and hence doesn't concern the IPoIB discussions. The actual VL that a packet uses is configured by the SM in the switch/channel adapter tables and is determined based on the Service Level (SL) specified in every packet. There are 16 possible SLs.

In addition to the features described above viz. Queue Pairs(QPs), Service Levels(SLs) and addressing(GID/LID), IBA

also defines the following:

Partitioning:

> Every packet, but for the raw datagrams, carries the
> partition key (P_key). These values are used for
> isolation in the fabric. A switch (this is an optional
> feature) may be programmed by the SM to drop packets
> not having a certain key. The CA ports always check
> for the P_Keys. A CA port may belong to multiple
> partitions. P_Key checking is optional at IB routers.

> A P_Key may be described as having 'limited
> membership' or 'full membership'. For a packet to be
> accepted at least one of the P_Keys i.e. the P_Key in
> the packet or the P_Key in the port, must be 'full
> membership' P_Keys.

Q_Keys:

> Q_Keys are used to enforce access rights for reliable
> and unreliable IB datagram services. Raw datagram
> services don't use Q_Keys. At communication
> establishment the endpoints exchange the Q_Keys and
> must always use the relevant Q_Keys when communicating
> with one another. Multicast packets use the Q_Key
> associated with the multicast group.

> Q_Keys with the most significant bit set are
> considered controlled Q_Keys (such as the GSI Q_Key)
> and a HCA does not allow a consumer to arbitrarily
> specify a controlled Q_Key. An attempt to send a
> controlled Q_Key results in using the Q_Key in the QP
> context. Thus the OS maintains control since it can
> configure the QP context for the controlled Q_Key for
> privileged consumers. It must be noted that though the
> notion of a 'controlled Q_Key' is suggested by IB
> specification it does not require its use or
> implementation.

Multicast support:

> A switch may support multicasting i.e. replication of
> packets across multiple output ports. This is an
> optional feature. Similarly, support for
> sending/receiving multicast packets is optional in
> CAs. A multicast group is identified by a GID. The GID
> format is as defined in [RFC2373] on IPv6 addressing.

Thus from an IPv6 over InfiniBand's point of view the
data link multicast address looks like the network
address. An IB port must explicitly join a multicast
group by sending a request to the SM to receive
multicast packets. A port may send packets to any
multicast group. In both cases the multicast LID to be
used in the packets is received from the SM.

There are 6 methods for data transfer in IB architecture.
These are :

  1. Unreliable Datagram (unacknowledged - connectionless)

     The UD service is connectionless and unacknowledged.
     It allows the QP to communicate with any unreliable
     datagram QP on any node.

     The switches and hence each link can support only a
     certain MTU. The MTU ranges are 256 bytes, 512 bytes,
     1024 bytes, 2048 bytes, 4096 bytes. A UD packet cannot
     be larger than the smallest link MTU between the two
     peers.

  2. Reliable Datagram   (acknowledged - multiplexed)

     The RD service is multiplexed over connections between
     nodes called End to end contexts (EEC) which allows
     each RD QP to communicate with any RD QP on any node
     with an established EEC. Multiple QPs can use the same
     EEC and a single QP can use multiple EECs (one for
     each remote node per reliable datagram domain).

  3. Reliable Connected (acknowledged - connection oriented)

     The RC service associates a local QP with one and only
     one remote QP. The message sizes maybe as large as
     2^31 bytes in length. The CA implementation takes care
     of segmentation and assembly.

  4. Unreliable Connected (unacknowledged - connection oriented)

     The UC service associates one local QP with one and
     only one remote QP. There is no acknowledgment and
     hence no resend of lost or corrupted packets. Such
     packets are therefore simply dropped. It is similar to
     RC otherwise.

  5. Raw Ethertype (unacknowledged - connectionless)

The Ethertype raw datagram packet contains a generic
transport header that is not interpreted by the CA but
it specifies the protocol type. The values for
ethertype are the same as defined in RFC1700 for
ethertype.

6. Raw IPv6 (unacknowledged - connectionless)

Using IPv6 raw datagram service, the IBA CA can
support standard protocol layers atop IPv6 (such as
TCP/UDP). Thus native IPv6 packets can be bridged into
the IBA SAN and delivered directly to a port and to
its IPv6 raw datagram QP.

The first 4 types are referred to as IB transports. The latter
two are classified as Raw datagrams. There is no indication of
the QP number in the raw datagram packets. The raw datagram
packets are limited by the link MTU in size.

The two connected modes and the reliable datagram mode may
also support 'Automatic Path Migration(APM)'. This is an
optional facility that provides for a hardware based path
fail over. An alternate path is associated with the QP when
the connection/EE context is first created. If unrecoverable
errors are encountered the connection switches to using the
alternate path.

1.2.1 InfiniBand Addresses

The InfiniBand architecture borrows heavily from the IPv6
architecture in terms of the InfiniBand subnet structure and
global identifiers (GIDs).

The InfiniBand architecture defines the GID associated with a
port as a 128-bit unicast or multicast identifier. IBA derives
the GID address format from the IPv6 format[RFC_2373] with
some additional properties/restrictions defined to facilitate
efficient discovery, communication and routing.

Note:
The IBA refers to [RFC_2373] explicitly. It must be noted
that IBA is therefore unaffected by any further changes
that are introduced in IPv6 addressing architecture.

IBA defines two types of GIDs:
unicast and,
multicast.

1.2.1.1 **Unicast GIDs**

The unicast GIDs are defined, as in IPv6, with three scopes.
The IB specification states:

a. link local: This is defined to be FE80/10.

        The IB routers will not forward packets with a
        link local address in source or destination
        beyond the IB subnet.

b. site local: FEC0/10

        A unicast GID used within a collection of
        subnets which is unique within that collection
        (e.g. a data center or campus) but is not
        necessarily globally unique. IB routers must
        not forward any packets with either a
        site-local Source GID or a site-local
        Destination GID outside of the site.

c. global:
        A unicast GID with a global prefix, i.e. an IB
        router may use this GID to route packets
        throughout an enterprise or internet.

1.2.1.2 **Multicast GIDs**

The multicast GIDs also parallel the IPv6 multicast addresses.
The IB specification defines the multicast GIDs as follows:

    FFxy:<112 bits>

    Flag bits:

        The nibble, denoted by x above, are the 4 flag bits: 000T.

        The first three bits are reserved and are set to zero. The
        last bit is defined as follows:

            T=0: denotes a permanently assigned i.e. well known GID
            T=1: denotes a transient group

    Scope bits:

        The 4 bits, denoted by y in the GID above, are the scope
        bits. These scope values are described in Table 1.

scope value	Address value
0	Reserved
1	Unassigned
2	Link-local
3	Unassigned
4	Unassigned
5	Site-local
6	Unassigned
7	Unassigned
8	Organization-local
9	Unassigned
0xA	Unassigned
0xB	Unassigned
0xC	Unassigned
0xD	Unassigned
0xE	Global
0xF	Reserved

Table 1

The IB specification further refers to [RFC 2373] and
[RFC 2375] while defining the well known multicast addresses.
However, it then states that the well known addresses apply to
IB raw IPv6 datagrams only. It must be noted though that a
multicast group can be associated with only a single MGID.
Thus the same MGID cannot be associated with the UD mode and
the raw datagram mode.

1.3   InfiniBand Multicast Group Management

IB multicast groups, identified by Multicast Global
Identifiers (MGIDs), are managed by the subnet manager(SM).
The SM explicitly programs the IB switches in the fabric to
ensure that the packets are received by all the members of the
multicast group that request the reception of packets. SM also
needs to program the switches such that packets transmitted to
the group by any group member reach all receivers in the
multicast group.

IBA distinguishes between multicast senders and receivers.
Though all members of a multicast group can transmit to the
group (and expect their packets to be correctly forwarded) not
all members of the group are receivers. A port needs to
explicitly request that multicast packets addressed to the
group be forwarded to it.

A multicast group is created by sending a join request to the

SM. As will be explained later, IBA defines multiple modes for joining a multicast group. The subnet manager records the group's multicast GID and the associated characteristics. The group characteristics are defined by the group path MTU, whether the group will be used for raw datagrams or unreliable datagrams, the service level, the partition key associated with the group, the Local Identifier(LID) associated with the group etc. These characteristics are defined at the time of the group creation. The interested reader may lookup the 'MCMemberRecord' attribute in the IB architecture specification[IB_ARCH] for the complete list of characteristics that define a group.

A LID is associated with the multicast group by the subnet manager(SM) at the time of the multicast group creation. The SM determines the multicast tree based on all the group members and programs the relevant switches. The Multicast LID(MLID) is used by the switches to route the packets.

Any member IB port wanting to participate in the multicast group must join the group. As part of the join operation the node receives the group characteristics from the SM. At the same time the subnet manager ensures that the requester can indeed participate in the group by verifying that it can support the group MTU, and accessibility to the rest of the group members. Other group characteristics may need verification too.

The SM, for groups that span IB subnet boundaries, must interact with IB routers to determine the presence of this group in other IB subnets. If present the MTU must match across the IB subnets.

P_Key is another characteristic that must match across IB subnets since the P_Key inserted into a packet is not modified by the IB switches or IB routers. Thus if the P_Keys didn't match the IB router(s) itself might drop the packets or destinations on other subnets might drop the packets.

A join operation may cause the SM to reprogram the fabric so that the new member can participate in the multicast group. By the same token a leave may cause the SM to reprogram the fabric to stop forwarding the packets to the requester.

1.3.1 Multicast Member Record

The multicast group is maintained by the SM with each of the group members represented by an MCMemberRecord[IB_ARCH]. Some

of its components are:

```
MGID        - Multicast GID for this multicast group
PortGID     - Valid GID of the port joining this multicast group
Q_Key       - Q_Key to be used by this multicast group
MLID        - Multicast LID for this multicast group
MTU         - MTU for this multicast group
P_Key       - Partition key for this multicast group
SL          - Service Level for this multicast group
Scope       - Same as MGID address scope
JoinState   - Join/Leave status requested by the port:
                bit 0: FullMember
                bit 1: NonMember
                bit 2: SendOnlyNonMember
```

1.3.1.1 JoinState

The JoinState indicates the membership qualities a port wishes
to add while joining/creating a group or delete when leaving a
group. The meaning of the JoinState bits are:

FullMember:
    Messages destined for the group are routed to and from
    the port. A group may be deleted by the SM if there
    are no FullMembers in the group.

NonMember:
    Messages destined for the group are routed to and from
    the port. The port is not considered a member for
    purposes of group creation/deletion.

SendOnlyNonMember:
    Group messages are only routed from the port but not
    to the port. The port is not considered a member for
    purposes of group creation/deletion.

A port may have multiple bits set in its record. In such case
the membership qualities are a union of the JoinStates. A port
may leave the multicast group for each of the JoinStates
individually or in any combination of JoinState
bits[IB_ARCH].

1.3.2 Join and Leave Operations

An IB port joins a multicast group by sending a join
request(SubnAdmSet() method) and leaves a multicast group by
sending a leave message (SubnAdmDelete() method) to the SM.
The IBA specification[IB_ARCH] describes the methods and

attributes to be used when sending these messages.

1.3.2.1 Creating a Multicast Group

There is no 'create' command to form a new multicast group.
The FullMember bit in the JoinState must be set to create a
multicast group. In other words, the first FullMember join
request will cause the group to be created as a side effect of
the join request. Subsequent join or leave requests may
contain any combination of the JoinState bits.

The creator of the group specifies the Q_Key, MTU, P_Key, SL,
FlowLabel, TClass and the Scope value. A creator may request
that a suitable MGID be created for it. Alternatively, the
request can specify the desired MGID. In both cases the MLID
is assigned by the SM.

Thus a group will be created with the specified values when
the requester sets the FullMember bit and no such group
already exists in the subnet.

1.3.2.3 Deleting a Multicast Group

When the last FullMember leaves the multicast group the SM may
delete the multicast group releasing all resources, including
those that might exist in the fabric itself, associated with
the group.

Note that a special 'delete' message does not exist. It is a
side effect of the last FullMember 'leave' operation.

1.3.2.4 Multicast Group Create/Delete Traps

The SA may be requested by the ports to generate a report
whenever a multicast group is created or deleted. The port can
specify the multicast group it is interested in i.e. use a
specific MGID or use a wild card request. The SA will report
these events using traps 66 (for creates) and 67 (for
deletes)[IB_ARCH].

Therefore, a port wishing to join a group but not create it by
itself may request a create notification or a port might even
request a notification for all groups that are created(a
wild card request). The SA will diligently inform them of the
creation utilizing the aforementioned traps. The requester can
then join the multicast group indicated. Similarly, a
SendOnlyNonMember or a NonMember might request the SA to
inform it of group deletions. The endnode, on receiving a

delete report, can safely release the resources associated
with the group. The associated MLID is no longer valid for the
group and may be reassigned to a new multicast group by the
SM.

2.0 **Management of InfiniBand Subnet**

To aid in the monitoring and configuration of InfiniBand
subnet components a set of MIB modules need to be defined.
MIB modules are needed for the channel adapters, InfiniBand
interfaces, InfiniBand subnet manager, InfiniBand subnet
management agents and to allow the management of specific
device properties. It must be noted that the management
objects addressed in the IPoIB documents are for all of the
IB subnet components and are not limited to IP(over IB).
The relevant MIB modules are described in separate
documents and are not covered here.

3.0 **IP over IB**

As described in section 1.0, the InfiniBand architecture
provides a broad set of capabilities to choose from when
implementing IP over InfiniBand networks.

The IPoIB specification must not, and does not, require
changes in IP and higher layer protocols. Nor does it mandate
requirements on IP stacks to implement special user level
programs. It is an aim of IPoIB specification that the IPoIB
changes be amenable to modularization and incorporation into
existing implementations at the same level as other media
types.

3.1 **InfiniBand as Datalink**

InfiniBand architecture provides multiple methods of data
exchange between two endpoints as was noted above. These are:

        Reliable Connected (RC)
        Reliable Datagram (RD)
        Unreliable Connected (UC)
        Unreliable Datagram (UD)
        Raw Datagram : Raw IPv6 (R6)
                     : Raw Ethertype (RE)

IPoIB can be implemented over any, multiple or all of these
services. A case can be made for support on any of the
transport methods depending on the desired features.

The IB specification requires Unreliable Datagram mode to be supported by all the IB nodes. The host channel adapters(HCAs) are specifically required to support Reliable connected(RC) and Unreliable connected(UC) modes but the same is not the case with target channel adapters(TCAs). Support for the two Raw Datagram modes is entirely optional. The Raw Datagram mode supports a 16-bit CRC as against the better protection provided by the use of a 32-bit CRC in other modes.

For the sake of simplicity, ease of implementation and integration with existing stacks, it is desirable that the fabric support multicasting. This is possible only in Unreliable datagram (UD) and IB's Raw datagram modes.

Thus it is only the UD mode that is universal, supports multicast, and a robust CRC. Given these conditions it is the obvious choice for IP over InfiniBand [IPOIB_ENCAP].

Future documents might consider the connected modes. In contrast to the limited link MTU offered by UD mode, the connected modes can offer significant benefit in terms of performance by utilizing a larger MTU. Reliability is also enhanced if the underlying feature of automatic path migration of connected modes is utilized.

3.2 Multicast Support

InfiniBand specification makes support of multicasting in the switches optional. Multicast however, is a basic requirement in IP networks. Therefore, IPoIB requires that multicast capable InfiniBand fabrics be used to implement IPoIB subnets.

3.2.1 Mapping IP Multicast to IB Multicast

Well known IP multicast groups are defined for both IPv4 and IPv6 (RFC_1700, RFC_2373). Multicast groups may also be dynamically created at any time. To avoid creating unnecessary duplicates of multicast packets in the fabric, and to avoid unnecessary handling of such packets at the hosts each of the IP multicast groups needs to be associated with a different IB multicast group as far as possible. A process is defined in [IPOIB_ENCAP] for mapping the IP multicast addresses to unique IB multicast addresses.

3.2.2 Transient Flag in IB MGIDs

The IB specification describes the flag bits as discussed in

section 1.3. The IB specification also defines some well known
IB multicast GIDs(MGIDs). The MGIDs are reserved for the IB's
Raw datagram mode which is incompatible with the other
transports of IB. Any mapping that is defined from IP
multicast addresses therefore must not fall into IB's
definition of a well-known address.

Therefore all IPoIB related multicast GIDs always set the
transient bit.

3.3 IP Subnets Across IB Subnets

Some implementations may wish to support multiple clusters of
machines in their own IB subnets but otherwise be part of a
common IP subnet. For such a solution the IB specification
needs multiple upgrades. Some of the required enhancements
are:

1) A method for creating IB multicast GIDs that span multiple
   IB subnets. The partition keys and other parameters need to
   be consistent across IB subnets.

2) Develop IB routing protocol to determine the IB topology
   across IB subnets.

3) Define the process and protocols needed between IB nodes
   and IB routers

Until the above conditions are met it is not possible to
implement IPoIB subnets that span IB subnets. The IPoIB
standards have however been defined with this possibility in
mind.

4.0 IP Subnets in InfiniBand Fabrics

The IPoIB subnet is overlaid over the IB subnet. The IPoIB
subnet is brought up in the following steps:

Note: the join/leave operation at the IP level will be
      referred to as IP_join/IP_leave and the join/leave
      operations at the IB level will be referred to as
      IB_join in this document.

1. The all-IPoIB nodes IB multicast group is created

   The fabric administrator creates a IB multicast
   group(henceforth called 'broadcast group') when the IP subnet
   is setup. The 'broadcast group' is defined in [IPOIB_ENCAP].

The method by which the broadcast group is setup is not
defined by IPoIB. The group may be setup at the SM by the
administrator or by the first IB_join.

As noted earlier, at the time of creating an IB multicast
group, multiple values such as the P_Key, Q_Key, Service
Level, Hop Limit, Flow ID, TClass, MTU etc., have to be
specified. These values should be such that all potential
members of the IB multicast group are be able to communicate
with one another when using them. In the future, as the IB
specification associates more meaning with the various
parameters and defines IB QoS, different values for IP
multicast traffic may be possible. All unicast packets also
need to use the P_Key and Q_Key specified in the broadcast
group [IPOIB_ENCAP]. It is obvious that a thought out
configuration is required for a successful setup of the IPoIB
subnet.

2. All IPoIB interfaces IB_join the broadcast group

The broadcast group defines the span and the members of the
IPoIB link. This link gets built up as IPoIB nodes IB_join the
broadcast group.

The IB_join to the broadcast group has the additional benefit
of distributing the above mentioned multicast group parameters
to all the members of the subnet.

Note that this IB_join to the broadcast group is a FullMember
join. If any of the ports or the switches linking the port to
the rest of the IPoIB subnet cannot support the
parameters(e.g. path MTU or P_Key) associated with the
broadcast group, then the IB_join request will fail and the
requesting port will not become part of the IPoIB subnet.

3. Configuration Parameters

As noted above, parameters such as, Q_Key, Path MTU, needed
for all IPoIB communication are returned to the IPoIB node on
IB_joining the 'broadcast group'. [IPOIB_ENCAP] also notes
that the parameters used in the broadcast group are used when
creating other multicast groups.

However, the P_Key must still be known to the IPoIB endnode
before it can join the broadcast-group. The P_Key is included
in the mapping of the broadcast group[IPOIB_ENCAP]. Another
parameter, the scope of the broadcast group, also needs to be
known to the endnode before it can join the broadcast group.

It is an implementation choice on how the P_Key and the scope
bits related to the IPoIB subnet are determined by the
implementation. These could be configuration parameters
initialized by some means by the administrator.

The methods employed by an implementation to determine the
P_Key and scope bits are not specified by IPoIB.

4.1 IPoIB VLANs

The endpoints in an IB subnet must have compatible P_Keys to
communicate with one another. Thus the administrator when
setting up an IP subnet over an IB subnet must ensure that all
the members have compatible P_Keys. An IP subnet can have only
one P_Key associated with it to ensure that all IP nodes in it
can talk to one another. An endpoint may however have multiple
P_Keys.

The IB architecture specifies that there can be only one MGID
associated with a multicast group in the IB subnet. The P_Key
is included in the MGID mappings from the IP multicast
addresses[IPOIB_ENCAP]. Since the P_Key is unique in the IB
subnet the inclusion of the P_Key in the IB MGIDs ensures that
unique MGID mappings are created. Every unique broadcast group
MGID so formed creates a separate abstract IPoIB link and
hence an IPoIB VLAN.

4.2 Multicast in IPoIB subnets

IP multicast on InfiniBand subnets follows the same concepts
and rules as on any other media. However, unlike most other
media multicast over InfiniBand requires interaction with
another entity, the IB subnet manager. This section describes
the outline of the process and suggests some guidelines.

IB architecture specifies the following format for IB
multicast packets when used over unreliable datagram(UD)
mode:

```
+--------+-------+---------+---------+-------+---------+---------+
|Local   |Global |Base     |Datagram |Packet |Invariant| Variant |
|Routing |Routing|Transport|Extended |Payload| CRC     |  CRC    |
|Header  |Header |Header   |Transport| (IP)  |         |         |
|        |       |         |Header   |       |         |         |
+--------+-------+---------+---------+-------+---------+---------+
```

For details about the various headers please refer to
InfiniBand Architecture Specification[IB_ARCH].

The Global routing header (GRH) includes the IB multicast group GID. The Local routing header (LRH) includes the local identifier (LID). The IB switches in the fabric route the packet based on the LID.

The GID is made available to the receiving IB user (the IPoIB interface driver for example). The driver can therefore determine the IB group the packet belongs to.

IPv4 defines three levels of multicast compliance. These are:

   Level 0: No support for IP multicasting

   Level 1: Support for sending but not receiving multicasts

   Level 2: Full support for IP multicasting

In IPv6 there is no such distinction. Full multicast support is mandatory. Additionally, all IPv4 subnets support broadcast(255.255.255.255). IPv4 broadcast can always be sent/received by all IPv4 interfaces.

Every IPoIB subnet requires the broadcast GID to be defined. Thus a packet can always be broadcast.

4.2.1 Sending IP Multicast Datagrams

An IP host may send a multicast packet at any time to any multicast address.

The IP layer conveys the multicast packet to the IPoIB interface driver/module. This module attempts to IB_join the relevant IB multicast group. This is required since otherwise InfiniBand architecture does not guarantee that the packet will reach its destinations.

A pure sender may choose to join the multicast group as a FullMember. In such a case the sender will receive all the multicast packets transmitted to the IB group. Additionally, the IB group will not be deleted until the sender leaves the group.

Alternatively, a sender might IB_join as a SendOnlyNonMember. In such a case the packets are not routed to the sender though packets transmitted by it can reach the other group members. Additionally, the group can be deleted when all FullMembers have left the group. The sender can further request delete updates from the SM.

If the sender does not find the group in existence it is
recommended in [IPOIB_ENCAP] that the packets be sent to the
MGID corresponding to the all-IP routers address. A sender
could also send the packets to the broadcast group. The
sender might also choose to request 'creation' reports from
the SM.

4.2.2 Receiving Multicast Packets

The IP host must join the IB multicast group corresponding to
the IP address. This follows from the IBA requirement that the
receiver must join the relevant IB multicast group. The group
is automatically created if it does not exist [IB_ARCH].

The IP receivers must IB_leave the IB group when the IP layer
stops listening of the corresponding IP address. The SM can
then choose to delete the group.

4.2.3 Router considerations for IPoIB

IP routers know of the new IP groups created in the subnet by
the use of protocols such as IGMP/MLD. However, this is not
enough for IPoIB since the router needs to IB_join the
relevant IB groups to be able to receive and transmit the
packets. There is no promiscuous mode for listening to all
packets.

The IPoIB routers therefore need to request the SM to report
all creations of IB groups in the fabric. The IPoIB router can
then IB_join the reported group. It is not desirable that the
router's IB_joining of a multicast group be considered the
same as the IB_join from a receiver - the router's IB_join
shouldn't disallow the group's deletion when all receivers
leave. To overcome just this type of situations, IBA provides
the NonMember IB_join mode.

The NonMember IB_join mode can be used by IP routers when they
join in response to the create reports. A router should
ideally request the delete reports too so that it can release
all the resources associated with the group. The MLID
associated with a deleted MGID can be reassigned by the SM and
therefore there is a possibility of erroneous transmissions if
the MLID is cached. A router that does not request delete
reports will still work correctly since it will receive the
correct MLID , and purge any old cached value, when it
IB_joins the IB group in response to a create report.

It is reasonable for a router to IB_join as a FullMember if it

is joining the IB group in response to an application/routing
daemon request. In such a case the router might end up
controlling the existence of the IB group (since it is a
FullMember of the group).

4.2.4 Impact of InfiniBand Architecture Limits

An HCA or TCA may have a limit on the number of MGIDs it can
support. Thus, even though the groups may not be limited at
the subnet manager and in the subnet as such, they may be
limited at a particular interface. It is advisable to choose
an adequately provisioned HCA/TCA when setting up an IPoIB
subnet.

4.2.5 Leaving/Deleting a Multicast Group

An IPv4 sender (level 1 compliance) IB_joins the IB multicast
group only because that is the only way to guarantee reception
of the packets by all the group recipients. The sender must
however IB_leave the group at some time. A sender could, when
not a receiver on the group, start a timer per multicast group
sent to. The sender leaves the IB group when the timer goes
off. It restarts the timer if another message is sent.

This suggestion doesn't apply to the IB broadcast group. It
also doesn't apply to the IB group corresponding to the
all-hosts multicast group. An IPv4 host must always remain a
member of the broadcast group.

An IP multicast receiver IB_leaves the corresponding IB
multicast group when it IP_leaves the IP multicast group. In
the case of IPv4 implementation the receiver may choose to
continue to be a sender (level 1 compliance). In which case it
may choose not to IB_leave the IB group but start a timer as
explained above.

As noted elsewhere, the SM can choose to free up the
resources(e.g. routing entries in the switches) associated
with the IB group when the last FullMember IB_leave the group.
The MLID therefore becomes invalid for the group. The MLID can
be reassigned when a new group is created.

SendOnlyNonMember/NonMember ports caching the MLID need to
avoid this possibility. The way out is for them to request
group delete reports. An IP router requesting reports for all
groups need not request the delete report since an IB_join in
response to a create report will return the new MLID
association to it.

A router might prefer to IB_leave the IB multicast group when
there are no members of the IP multicast address in the subnet
and it has no explicit knowledge of any need to forward such
packets.

4.3 Transmission of IPoIB packets

The encapsulation of IP packets in InfiniBand is described
in[IPOIB_ENCAP].

It specifies the use of an 'Ethertype' value [IANA] in all
IPoIB communication packets. The link-layer address is
comprised of the Global Identifier(GID) and the Queue Pair
Number(QPN) [IPOIB_ENCAP].

To allow for multiple IB subnet based IPoIB subnets, the
specification utilizes the Global Identifier(GID) as part of
the link-layer address. Since all packets in IB have to use
the Local Identifier(LID) the address resolution process has
the additional step of resolving the destination GID, returned
in response to ARP/ND request, to the LID[IPOIB_ENCAP]. This
phase of address resolution might also be used to determine
other essential parameters (e.g. the SL, path rate etc.)for
successful IB communication between two peers.

As noted earlier, all communication in the IPoIB subnet
derives the Q_Key to use from the Q_Key specified in the
broadcast group.

4.4 RARP and Static ARP entries

RARP entries or static ARP entries are based on invariant
link-addresses. In the case of IPoIB, the link-address
includes the QPN which might not be constant across reboots or
even across network interface resets. Therefore, static ARP
entries or RARP server entries will only work if the
implementation(s) using these options can ensure that the QPN
associated with an interface is invariant across
reboots/network resets[IPOIB_ENCAP].

4.5 DHCPv4 and IPoIB

DHCPv4 [RFC_2131] utilizes a 'client identifier' field
(expected to hold the link-layer address) of 16 bytes. The
address in the case of IPoIB is 20-bytes. To get around this
problem IPoIB specifies [IPOIB_DHCP] that the 'broadcast flag'
be used by the client when requesting an IP address.

5.0 QoS and Related Issues

The IB specification suggests the use of service levels for
load balancing, QoS and deadlock avoidance within an IB
subnet. But the IB specification leaves the usage and mode of
determination of the SL for the application to decide. The SL
and list of SLs are available in the SA but it is up to the
endnode's application to choose the 'right' value.

Every IPoIB implementation will determine the relevant SL
value based on its own policy. No method or process for
choosing the SL has been defined by the IPoIB standards.

6.0 Security Considerations

This document describes the IB architecture as relevant to
IPoIB. It further restates issues specified in other
documents. It does not itself specify any requirements. There
are no security issues introduces by this document. IPoIB
related security issues are described in [IPOIB_ENCAP] and
[IPOIB_DHCP].

7.0 Acknowledgments

This document has benefited from the comments and suggestions
of the members of the IPoIB working group and the members of
the InfiniBand(SM) Trade Association.

8.0 References

8.1 Normative References

[IB_ARCH]   InfiniBand Architecture Specification, Volume 1.1

[IPOIB_ENCAP] draft-ietf-ipoib-ip-over-infiniband-06.txt

[IPOIB_DHCP]   draft-ietf-ipoib-dhcp-over-infiniband-05.txt

8.2 Informative References

[RFC_2373]   IP Version 6 Addressing Architecture

[RFC_2375]   IPv6 Multicast Address Assignments

[RFC_1700]   Assigned Numbers

[RFC_1112]   Host extensions for IP multicasting

[RFC_2236]   Internet Group Management Protocol, Version 2

[RFC_2710]   Multicast Listener Discovery

9.0 **Author's Address**

Vivek Kashyap

IBM
15450, SW Koll Parkway
Beaverton, OR 97006

Phone: +1 503 578 3422
Email: vivk@us.ibm.com

Full Copyright Statement