X3T11/96-
X3T11/Project xxx/REV 0.1

HIGH-PERFORMANCE PARALLEL INTERFACE - Multiple Path

(HIPPI-MP)

working draft proposed
American National Standard
for Information Systems

February 21, 1996

Secretariat:

Information Technology Industry Council (ITI)

ABSTRACT: This standard specifies the formats and procedures for the segmentation and reassembly of large upper-layer protocol data units for transmission over multiple High-Performance Parallel Interface – Framing Protocol (HIPPI-FP) lower-layer protocol instances. Single HIPPI interfaces support 800 or 1600 Mbit/s data transfer rates; HIPPI-MP allows multiple interfaces to be ganged together for even higher data transfer rates.

NOTE:

This is an internal working document of X3T11, a Technical Committee of Accredited Standards Committee X3. As such, this is not a completed standard. The contents are actively being modified by X3T11. This document is made available for review and comment only. For current information on the status of this document contact the individuals shown below:

POINTS OF CONTACT:

Roger Cummings (X3T11 Chairman)
Distributed Processing Technology
140 Candace Drive
Maitland, FL 32751
(407) 830-5522 x348, Fax: (407) 260-5366
E-mail: cummings_roger@dpt.com

Carl Zeitler (X3T11 Vice-Chairman)
IBM Corporation, MS 9440
11400 Burnet Road
Austin, TX 78758
(512) 838-1797, Fax: (512) 838-3822
E-mail: zeitler@ausvm6.vnet.ibm.com

Don Tolmie (HIPPI-MP Technical Editor)
Los Alamos National Laboratory
CIC-5, MS-B255
Los Alamos, NM 87545
(505) 667-5502, Fax: (505) 665-7793
E-mail: det@lanl.gov

Contents

Page

Foreword ... ii

1 Scope ... 1

2 Normative references ... 1

3 Definitions and conventions ... 1
- **3.1** Definitions ... 1
- **3.2** Editorial conventions ... 1

4 System overview ... 2
- **4.1** Typical Source operations ... 3
- **4.2** Typical Destination operations ... 3

5 HIPPI-MP service interface to upper layers ... 3
- **5.1** Service primitives ... 4
- **5.2** Sequences of primitives ... 4
- **5.3** ULP data transfer service primitives ... 4
- **5.4** Control service primitives ... 7
- **5.5** Status service primitives ... 7

6 Chunk operations ... 8
- **6.1** Segmentation ... 8
- **6.2** MP_Header ... 8
- **6.3** Reassembly ... 9
- **6.4** Source-side errors ... 9
- **6.5** Destination-side errors ... 9

Figures

Figure 1 – System overview ... 2
Figure 2 – HIPPI-MP service interface ... 3
Figure 3 – Data transfer service primitives ... 4
Figure 4 – Control service primitives ... 7
Figure 5 – Status service primitives ... 7
Figure 6 – First HIPPI-MP chunk with HIPPI-FP header ... 10
Figure 7 – Second, or later, HIPPI-MP chunk with HIPPI-FP header ... 10

Foreword (This Foreword is not part of American National Standard X3.xxx-199x.)

This standard specifies the formats and procedures for the segmentation and reassembly of large upper-layer protocol data units for transmission over multiple High-Performance Parallel Interface - Framing Protocol (HIPPI-FP) lower-layer protocol instances. Single HIPPI interfaces support 800 or 1600 Mbit/s data transfer rates; HIPPI-MP allows multiple interfaces to be aggregated together for even higher data transfer rates.

Requests for interpretation, suggestions for improvement or addenda, or defect reports are welcome. They should be sent to the X3 Secretariat, Computer and Business Equipment Manufacturers Association, 1250 Eye Street, NW, Suite 200, Washington, DC 20005.

This technical report was processed and approved for submittal to ANSI by Accredited Standards Committee on Information Processing Systems, X3. Committee approval of the technical report does not necessarily imply that all committee members voted for approval. At the time it approved this technical report, the X3 Committee had the following members:

(List of X3 Committee members to be include included in the published standard by the ANSI Editor.)

Subcommittee X3T11 on Device Level Interfaces, which developed this technical report, had the following members:

(List of X3T11 committee members will be included in published standard)

X3 Technical Report for Information Technology

High-Performance Parallel Interface – Multiple Path (HIPPI-MP)

1 Scope

This American National Standard specifies the formats and procedures for the segmentation and reassembly of large upper-layer protocol data units for transmission over multiple HIPPI-FP lower-layer protocol instances. Single HIPPI-PH interfaces support 800 or 1600 Mbit/s data transfer rates; HIPPI-MP allows multiple interfaces to be ganged together for even higher data transfer rates.

2 Normative references

The following American National Standard contains provisions which, through reference in this text, constitute provisions of this American National Standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standard listed below.

ANSI X3.210-1992, *High-Performance Parallel Interface – Framing Protocol (HIPPI-FP).*

ANSI X3.222-1993, *High-Performance Parallel Interface – Physical Switch Control (HIPPI-SC).*

3 Definitions and conventions

3.1 Definitions

For the purposes of this American National Standard, the following definitions apply.

3.1.1 chunk: The protocol data unit, consisting of ULP data and MP_Header, passed from HIPPI-MP to a lower-layer HIPPI-FP instance.

3.1.2 connection control information (CCI): A parameter sent as part of the sequence of operations establishing a connection from a Source to a Destination.

3.1.3 Destination: The equipment at the end of the interface that receives the data.

3.1.4 protocol data unit (PDU): A formatted data set that is a logical unit of interchange between entities.

3.1.5 service interface: Connection points to the ULP.

3.1.6 Source: The equipment at the end of the interface that transmits the data.

3.1.7 ULP data set: The data transferred between the ULP and the HIPPI-MP.

3.1.8 upper-layer protocol (ULP): A protocol immediately above the HIPPI-MP service interface.

3.2 Editorial conventions

In this document, certain terms that are proper names of signals or similar terms are printed in uppercase to avoid possible confusion with other uses of the same words (e.g., FLAG). Any lowercase uses of these words have the normal technical English meaning.

A number of conditions, sequence parameters, events, states, or similar terms are printed with the first letter of each word in uppercase and the rest lowercase (e.g., State, Source). Any lowercase uses of these words have the normal technical English meaning.

4 System overview

Figure 1 shows an overview of a system using HIPPI-MP to segment and reassemble large PDUs from upper-layer protocols ULP-1 and ULP-2. Below the HIPPI-FP instances, the physical layer, shown as HIPPI-PH instances in figure 1, may be implemented as any of the following:

– ANSI X3.183, High-Performance Parallel Interface - Mechanical, Electrical, and Signalling Protocol Specification (HIPPI-PH), operating with either the 800 Mbit/s or 1600 Mbit/s options,

– ANSI X3.xxx, High-Performance Parallel Interface – Serial Specification (HIPPI-Serial),

– ANSI X3.254, Fibre Channel - Mapping to HIPPI-FP (FC-FP), or ,

– any other protocol that operates under HIPPI-FP.

The ULPs shown may also connect directly to the HIPPI-FP instances while also connecting to HIPPI-MP, as shown in the dotted line between ULP-1 and the HIPPI-FP instance. Other ULPs, e.g., ULP-3 in figure 1, may connect directly to a HIPPI-FP instance without connecting to HIPPI-MP.

Switches, or other systems operating at the physical level, may be between the Source and Destination physical layers, e.g., HIPPI switches conforming to ANSI X3.218, HIPPI-SC, between the HIPPI-PH instances in figure 1.

Determination of how many physical paths to use between the Source and Destination is outside the scope of this standard. Determination of which physical paths to use is outside the scope of this standard. How the physical connections are set up and torn down is outside the scope of this standard.

NOTE – A management entity may maintain a table of mapping information as to whether a Destination supports HIPPI-MP, how many physical paths it supports, and their addresses.

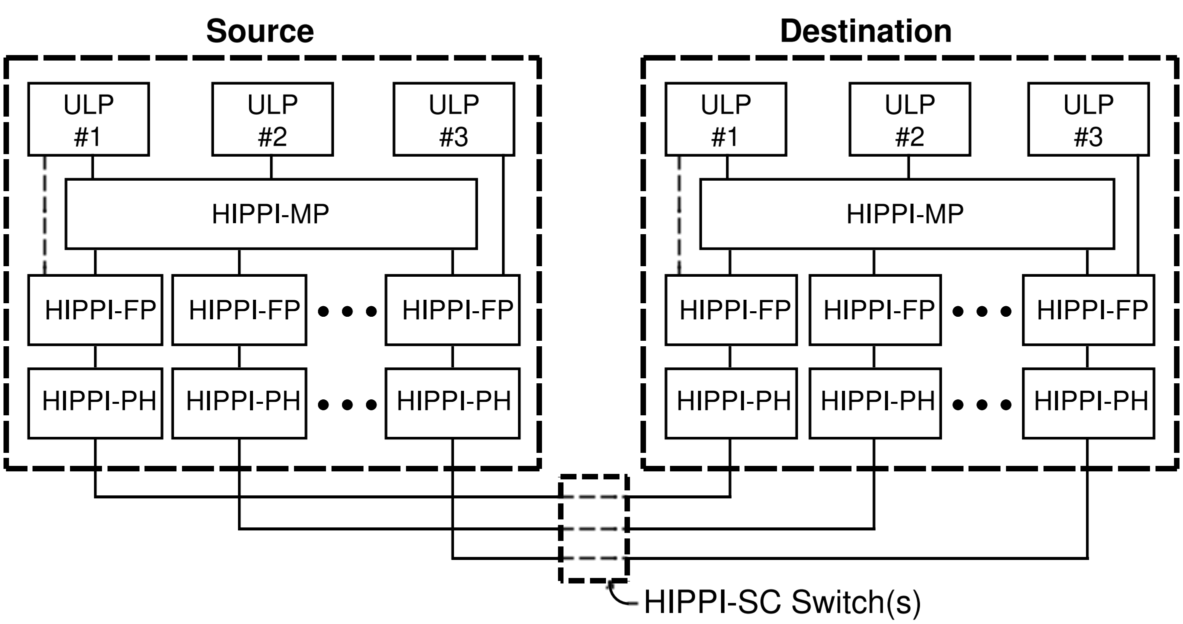

Figure 1 – System overview

4.1 Typical Source operations

On the Source side, the nominal steps in a transfer operation include:

a) The ULP passes its PDU to HIPPI-MP;

b) HIPPI-MP segments the PDU into smaller sized units, called chunks, builds an MP_Header for each chunk and attaches it to the chunk;

c) HIPPI-MP passes the chunks and MP_Headers to the lower-layer HIPPI-FP entities for transmission to the Destination;

d) HIPPI-FP appends a HIPPI-FP header to each chunk and passes the resultant HIPPI-FP packet to a lower-layer physical layer for transmission to the Destination. The packet format is as shown in figure 6.

4.2 Typical Destination operations

On the Destination side, the nominal steps in a transfer operation include:

a) The HIPPI-FP strips off the HIPPI-FP header, and since the ULP-id = HIPPI-MP in the HIPPI-FP header it passes the chunk to HIPPI-MP;

b) The Destination HIPPI-MP reassembles the ULP PDU associated with a particular ULP-id and Source-id, using the Sequence# to check for missing chunks;

c) Upon seeing the Last_Chunk bit, and receiving all of the chunks or timing out, HIPPI-MP passes the ULP PDU to the ULP identified by the ULP-id field in the MP_Header.

5 HIPPI-MP service interface to upper layers

This clause describes the services provided by HIPPI-MP. The intent is to provide the formalism necessary to relate this interface to other HIPPI interfaces. How many of the services described herein are chosen for a given implementation, and whether others may be required, is up to the implementer; however, a set of HIPPI-MP services must be supplied sufficient to satisfy the ULP(s) being used. The services as defined herein do not imply any particular implementation, or any interface.

In this standard the ULP and station management protocol (SMT) are service users, and the HIPPI-MP is the service provider to the ULP and SMT. The interfaces consist of the ULP primitives, prefixed with MP_, and the SMT primitives, prefixed with MPSM_.

The HIPPI-FP is also the service user of the HIPPI-FP services, prefixed with FP_. As might be expected, the HIPPI-MP service interface uses the same parameters and functions as the HIPPI-FP service interface.

Figure 2 shows the relationship of the HIPPI-MP interfaces.

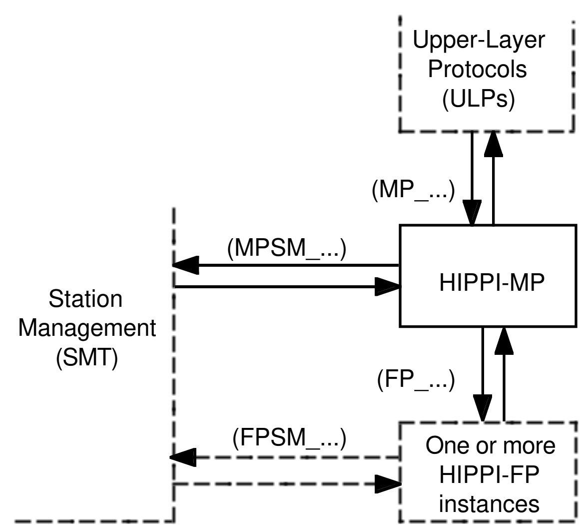

Figure 2 – HIPPI-MP service interface

5.1 Service primitives

All of the primitives and parameters are considered required except where explicitly stated otherwise.

HIPPI service primitives are of four types.

– *Request primitives* are issued by a service user to initiate a service from the service provider. In this standard, a second Request primitive of the same name shall not be issued until the Confirm for the first request is received.

– *Confirm primitives* are issued by the service provider to acknowledge a Request.

– *Indicate primitives* are issued by the service provider to notify the service user of a local event. This primitive is similar in nature to an unsolicited interrupt. Note that the local event may have been caused by a service Request. In this standard, a second Indicate primitive of the same name shall not be issued until the Response for the first Indicate is received.

– *Response primitives* are issued by a service user to acknowledge an Indicate.

5.2 Sequences of primitives

The order of execution of service primitives is not arbitrary. Logical and time sequence relationships exist for all described service primitives. Time sequence diagrams, as in figure 3, are used to illustrate a valid sequence. Other valid sequences may exist. The sequence of events between peer users across the user/provider interface is illustrated. In the time sequence diagrams the HIPPI-MP users are depicted on either side of the vertical bars while the service provider is in the center. A ULP or SMT implementation may present multiple requests for services, but the requests shall be serviced one at a time and in the order presented.

5.3 ULP data transfer service primitives

These primitives, as illustrated in figure 3, shall be used to transfer ULP data from the Source ULP to the Destination ULP.

5.3.1 ULP Identifiers

The ULP-id of the HIPPI-MP header designates the Destination ULP to which the data set is to be delivered. The ULP-id values are assigned in subclause 5.4.1 of ANSI X3.210, HIPPI-FP.

The ULP-id value for the ULP transferring data though HIPPI-MP will be carried in the MP_Header. The ULP value for HIPPI-MP will be in the FP_Header.

5.3.2 MP_TRANSFER.Request

Issued by the Source ULP to request a data transfer. The packet format is defined in 6.1 and 6.2, and shown in figures 6 and 7.

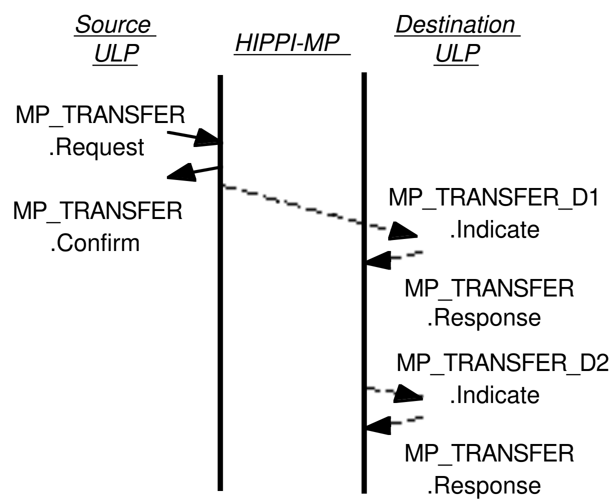

Figure 3 – Data transfer service primitives

Semantics –
MP_TRANSFER.Request
(CCI,
ULP-id,
D1_Size,
D1_Data_Set,
D2_Size,
D2_Data_Set,
Keep_Connection,
Start_D2_on_Burst_Boundary)

The CCI is the Connection Control Information pointing to the Destination. Note that this just defines a single physical path, and HIPPI-MP may use multiple physical paths.

The ULP-id identifies the Destination ULP. See 5.3.1.

D1_Size is the length, in bytes, of the ULP D1_Data_Set to be placed in the first burst of the packet. The maximum D1_Size shall be 1008 bytes. Note that the maximum D1_Size allowed by HIPPI-FP is 1016 bytes; the difference is the 8-byte HIPPI-MP header.

D1_Data_Set is the ULP data set to be placed in the first burst, and delivered separately from the D2_Data_Set. The D1_Data_Set is intended for control information.

D2_Size is the length, in bytes, of the ULP data set to be placed in the remainder of the packet. The maximum determinate D2_Size is 4,294,967,294 bytes (2^{32} - 2). A D2_Size of hexadecimal FFFFFFFF shall mean that the length is indeterminate at the start of the transfer. Indeterminate length packets may be longer or shorter than the maximum determinate size. An indeterminate length packet shall be broken up into a stream of determinate length chunks. The D2_Size shall be set to zero to indicate the absence of the D2_Data_Set.

D2_Data_Set is the ULP data set to be sent. Placement of the D2_Data_Set shall be governed by the Start_D2_on_Burst _Boundary parameter. The D2_Data_Set is intended for user data, or the whole data set if separate control information is not used.

Keep_Connection true says that another ULP data set with the same routing information is coming, and the physical HIPPI-PH connection should be maintained if possible. When Keep_Connection is false, the connection may be broken after this packet. It is recommended that this bit be kept false for HIPPI-MP to avoid locking up multiple paths.

Start_D2_on_Burst_Boundary controls the starting location for the D2_Area. If true, then the D2_Area shall start at the beginning of the second HIPPI-PH burst. If false, then the D2_Area may start in the first burst. This parameter is passed through HIPPI-MP to HIPPI-FP when transferring the first chunk.

Issued – The Source ULP issues this primitive to the Source HIPPI-FP to request the transfer of the ULP data set to the Destination.

Effect – The Source HIPPI-MP shall accept the ULP data set for transmission. HIPPI-MP shall segment the composite data set composed of the D1_Data_Set and D2_Data_Set into chunks, and pass each chunk to a HIPPI-FP instance. See 6.1 for chunk size suggestions.

The first chunk of the ULP's PDU, along with its MP_Header, shall be passed to a HIPPI-FP instance, with the following parameters:

CCI = as obtained from the ULP

ULP-id = HIPPI-MP (See 5.4.1 in HIPPI-FP.)

D1_Size = D1_Size from ULP + 8. The additional eight bytes are the MP_Header.

D1_Data_Set, = MP_Header and the D1_Data_Set from the ULP.

D2_Size = the number of bytes in the segment of the D2_Data_Set being transferred in this chunk.

D2_Data_Set = the segment of the D2_Data_Set being transferred in this chunk.

Keep_Connection = false

Start_D2_on_Burst_Boundary = as obtained from the ULP.

The second chunk, and following chunks, along with their MP_Headers, shall be passed to HIPPI-FP instances with the following parameters:

CCI = connection control information for a path to the Destination.

ULP-id = HIPPI-MP (See 5.4.1 in HIPPI-FP.)

D1_Size = 8

D1_Data_Set, = MP_Header

D2_Size = the number of bytes in this segment of the D2_Data_Set being transferred in this chunk.

D2_Data_Set = this segment of the D2_Data_Set being transferred in this chunk.

Keep_Connection = false

Start_D2_on_Burst_Boundary = false

5.3.3 MP_TRANSFER.Confirm

This primitive acknowledges the MP _TRANSFER.Request from the Source ULP.

Semantics –
MP_TRANSFER.Confirm

Issued – The HIPPI-MP shall issue this primitive to the Source ULP to acknowledge the MP_TRANSFER.Request.

Effect – Unspecified

5.3.4 MP_TRANSFER.Indicate

These primitives indicate to the Destination ULP that the D1_Data_Set, or D2_Data_Set, of a packet, addressed to this particular ULP has been received from the Source. HIPPI-MP shall not issue the MP_TRANSFER.Indicate primitive passing the D2_Data_Set until all of the D2_Data_Set chunks have been received or timed out.

Semantics –
MP_TRANSFER_D1.Indicate
(ULP-id,
CCI,
Status,
D2_Size,
D2_Offset,
D1_Area_Size,
D1_Area)

MP_TRANSFER_D2.Indicate
(ULP-id,
CCI, Status,
D2_Size,
D2_Offset,
D2_Data_Set)

ULP-id is the ULP to receive the data. See 5.3.1.

CCI is the CCI for the current connection.

Status denotes whether the data set being delivered was received with errors. Status includes, but is not limited to, errors in the packet.

D2_Size is the length of the D2_Data_Set, in bytes, as received in the FP_Header. If D2_Size equals hexadecimal FFFFFFFF, then it is up to the ULP to determine the validity and actual length of the D2_Data_Set.

D2_Offset is the number of unused bytes from the start of the D2_Area to the first byte of the D2_Data_Set. The D2_Offset is used by the Source and Destination to keep proper word alignment on the D2_Data_Set so as to avoid shifting and copying the data to achieve alignment at the Destination. The D2_Offset allows the Source memory image of the D2_Data_Set, even if it does not start on a 64-bit word boundary, to be reproduced at the Destination.

D1_Area_Size is the size of the D1_Area being passed to the ULP. The actual size of the D1_Data_Set is self defining within the D1_Area.

D1_Area contains the D1_Data_Set. It is up to the Destination ULP to determine the size of the D1_Data_Set and extract it from the D1_Area.

The D2_Data_Set is the D2 ULP data being delivered to the ULP.

Issued – The Destination HIPPI-MP may issue this primitive to pass the D1_Data_Set to the Destination ULP upon receipt of the first chunk and before arrival of the other chunks, or may wait for all of the chunks to arrive. The Destination HIPPI-MP shall issue this primitive to pass the D2_Data_Set to the Destination ULP when all of the chunks of a ULP data set have been received. A packet containing both the D1_Data_Set and the D2_Data_Set shall generate primitives for both the D1_Data_Set and the D2_Data_Set.

Effect – Unspecified

5.3.5 MP_TRANSFER.Response

This primitive acknowledges a MP_TRANSFER .Indicate for either the D1_Data_Set or the D2_Data_Set.

Semantics – MP_TRANSFER.Response

Issued – The Destination ULP issues this primitive to acknowledge receipt of the MP_TRANSFER.Indicate.

Effect – The Destination HIPPI-MP is enabled to issue another MP_TRANSFER.Indicate.

5.4 Control service primitives

These primitives, as illustrated in figure 4, shall be used to set parameters and control the interface. Note that a Control primitive can be initiated from either the Source or Destination.

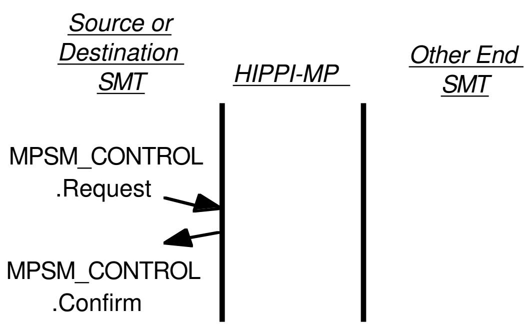

Figure 4 – Control service primitives

5.4.1 MPSM_CONTROL.Request

Issued by either the Source SMT or Destination SMT to set parameters, or otherwise control the local HIPPI-MP. Only one Source-side function is specified, others functions may be included in specific implementations.

Semantics –
MPSM_CONTROL.Request
(Command,
Command_Parameter)
The Command specifies the function to be performed. The parameters are specific to each function.

The Commands and Command_Parameters for the Source side include but are not limited to
Set CCI to physical link mappings

Issued – The Source or Destination SMT issues this primitive to perform some control function over the interface as a whole.

Effect – The HIPPI-MP shall perform the function specified.

5.4.2 MPSM_CONTROL.Confirm

This primitive acknowledges the MPSM _CONTROL.Request to the issuing SMT.

Semantics –
MPSM_CONTROL.Confirm (Status)

Status reports the success or failure of the MPSM_CONTROL.Request commands.

Issued – The HIPPI-MP shall issue this primitive to the SMT when the command specified in the MPSM_CONTROL.Request has been accepted.

Effect – Unspecified

5.5 Status service primitives

These primitives, as illustrated in figure 6, shall be used to obtain status information from the local HIPPI-MP. Note that a Status primitive can be initiated from either the Source or Destination, and shall only affect the local end of the interface.

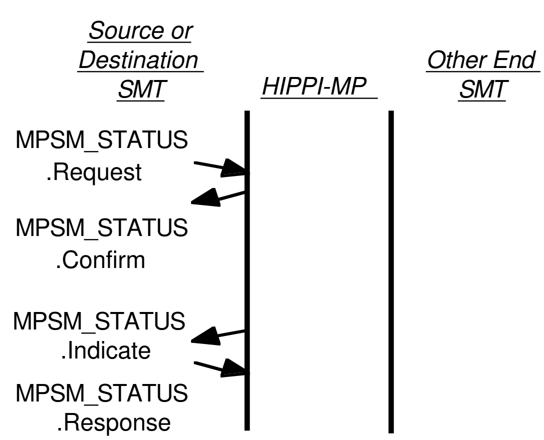

Figure 5 – Status service primitives

5.5.1 MPSM_STATUS.Request

Issued by either the Source SMT or Destination SMT to request a status report.

Semantics –
MPSM_STATUS.Request

Issued – The SMT issues this primitive when it wishes to obtain the status of the HIPPI-MP.

Effect – The HIPPI-FP shall respond with a MPSM_STATUS.Confirm.

5.5.2 FPSM_STATUS.Confirm

This primitive replies to the previous MPSM _STATUS.Request with status information.

Semantics –
MPSM_STATUS.Confirm (Status)

The Source side Status shall contain, but is not limited to
Errors
Current CCI to physical link mappings

The Destination side Status shall contain, but is not limited to
Errors

Issued – The HIPPI-MP shall issue this primitive to the SMT in response to a MPSM_STATUS .Request.

Effect – Unspecified

6 Chunk operations

6.1 Segmentation

The HIPPI-MP implementation shall break the ULP's PDU into chunks.

– The chunk size and placement shall be such that D2_Offset = 0 for the second and following chunks that contain D2_Data. D2_Offset may be ≠ 0 for the first chunk that contains D2_Data.

– The chunk size and placement shall be such that no Fill is used except on the last chunk.

NOTES

1 To enhance reassembly efficiency, it is recommended that the chunks be broken on natural memory boundaries, e.g., 1 KByte boundaries.

2 If the HIPPI-MP receives a small PDU for transmission, it will be more efficient to pass it as a single chunk rather than breaking it into even smaller chunks and incurring the overhead associated with the segmentation and reassembly processes.

3 The chunk size may be chosen to allow multiplexing through a HIPPI-SC switch. For example, by using a chunk size of 64 KBytes, and breaking the connection between chunks, other Sources will have an opportunity every 640 µs to interleave data transmissions to the common Destination link.

4 Highest efficiency will be obtained when the transmission time across all of the available links is equal, i.e., larger chunk sizes are used with higher speed links and smaller chunk sizes are used with lower speed links.

6.2 MP_Header

The Source HIPPI-MP instance shall prepend an MP_Header to all chunks before passing them to a HIPPI-FP instance. The MP_Header shall be eight bytes in length. For completeness, figures 6 and 7 show the resultant packets, for the first and following chunks, after the 8-byte HIPPI-FP header has been added, i.e., the MP_Header is words 2 and 3 in the figures.

NOTE – When using HIPPI-MP the maximum size of the D1_Data_Set is 1008 bytes, while it is 1016 bytes when using HIPPI-FP without HIPPI-MP. The difference is the 8-byte MP_Header.

The fields in the MP_Header shall be:

ULP-id = ULP-id of the upper layer protocol above HIPPI-MP that issued the PDU.

L = Last chunk (1 bit) = 1 designates that this is the last chunk of the ULP's PDU. L = 0 designates that there are more chunks in this PDU.

Rsv = Reserved (3 bits). All of the reserved bits shall be transmitted as zeros.

Source-id = ANSI X3.222, HIPPI-SC, logical address for this host (12 bits).

Sequence# = Chunk number of this piece of the ULP PDU. Sequence# shall start at zero and increment for each chunk. Sequence# shall wrap from FF to 0 if there are more than 256 chunks in the PDU.

D2_Address_Offset (32 bits) is the offset between the first word of the PDU, and the first word of this chunk.

6.3 Reassembly

When the ULP-id = HIPPI-MP in a received packet, then the receiving HIPPI-FP instance will strip off the HIPPI-FP header and pass the PDU to HIPPI-MP. HIPPI-MP shall use the values in the MP_Header to reassemble the individual chunk into its proper place in the ULP's PDU.

ULP-id = ULP-id of the ULP above HIPPI-MP that the data shall be sent to.

L = Last chunk (1 bit) = 1 indicates that this is the last chunk of the ULP's PDU. L = 0 indicates that there are more chunks in this PDU. HIPPI-MP shall verify that all chunks of a PDU have been received, or timed out, before signalling completion to the ULP with a MP_TRANSFER.Indicate primitive. All chunks received shall be determined by a linear set of received Sequence# with no gaps.

Rsv = Reserved (3 bits). The reserved bits shall be ignored.

Source-id = Logical address of the Source. HIPPI-MP may simultaneously receive packets from multiple Sources; the Source-id shall be used to differentiate between the packets. Note that HIPPI does not multiplex packets, i.e., between a Source and Destination there is only one packet in progress at a time; hence there is no need for a packet identifier in the MP_Header.

Sequence# = Chunk number of this piece of the ULP PDU.

D2_Address_Offset (32 bits) is the offset between the first word of the PDU, and the first word of this chunk. Implementations may use this value to move the chunk to the ULP's memory area before all of the chunks have been received.

6.4 Source-side errors

If the Source HIPPI-MP receives an error from a HIPPI-FP below it, e.g., connection rejected or timeout, then the Source HIPPI-MP shall transfer the affected chunk over a different path.

6.5 Destination-side errors

An error in the received data shall cause a MP_TRANSFER.Indicate, with the error cause noted in the Status field, to be issued to the ULP. HIPPI-MP does not do retransmission; retransmission is the responsibility of the ULP, i.e., HIPPI-MP does best-effort delivery with error notification to the receiving ULP.

6.5.1 Error detected at lower-layer

An error detected by the receiving HIPPI-FP, e.g., LLRC or parity error, shall result in the data received being passed to the ULP with Status indicating the error.

6.5.2 Chunk timeout

Missing chunks shall be indicated if after a timeout period the Sequence#s, of the chunks received for a ULP PDU, are not monotonically increasing from 0 with no gaps, or L = 0 on the chunk with the largest Sequence#. The chunks of this ULP PDU that are received shall be passed to the ULP with Status indicating a missing chunk. The default timeout period shall be one second, measured from receipt of the last chunk associated with this ULP PDU.

6.5.3 Error in MP_Header

If the Source-id value in the MP_Header is unknown to the Destination HIPPI-MP, then that chunk shall be discarded and its Sequence# ignored. This error will be noted by the ULP as a missing chunk.

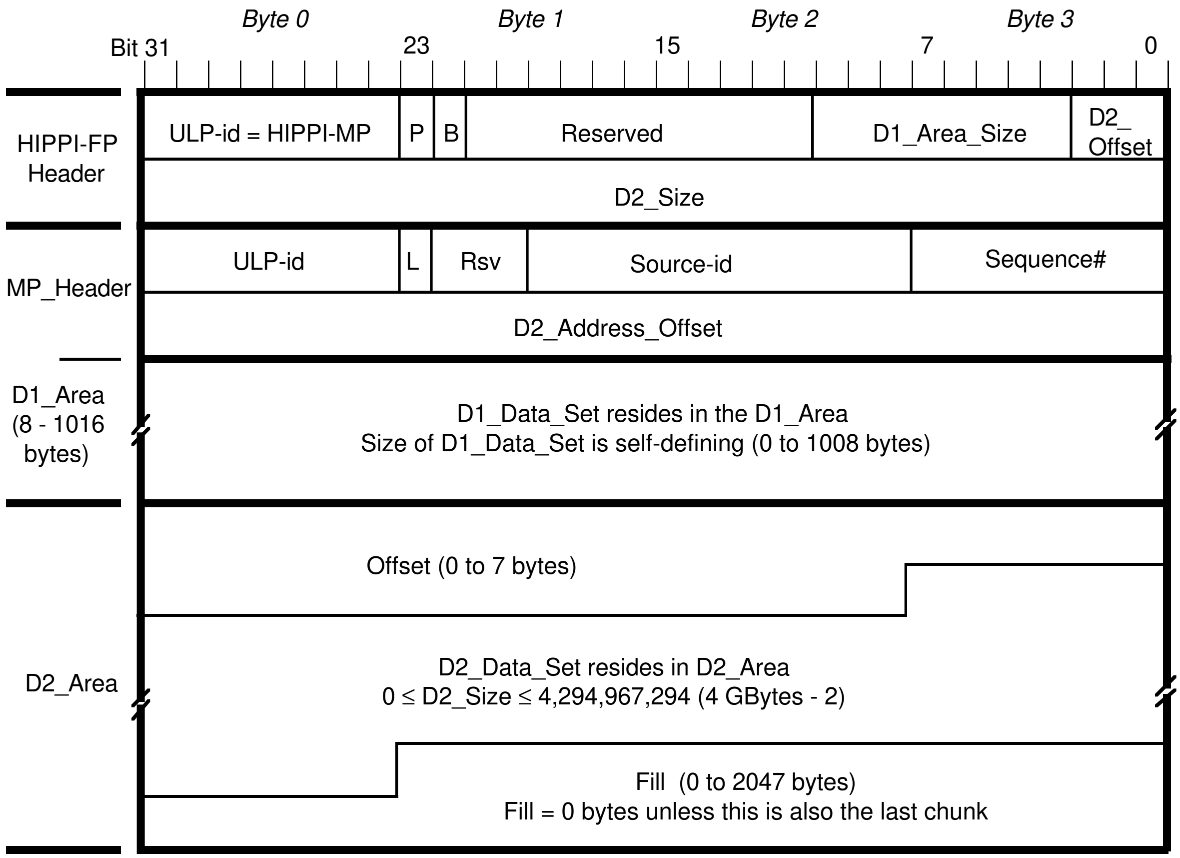

Figure 6 – First HIPPI-MP chunk with HIPPI-FP header

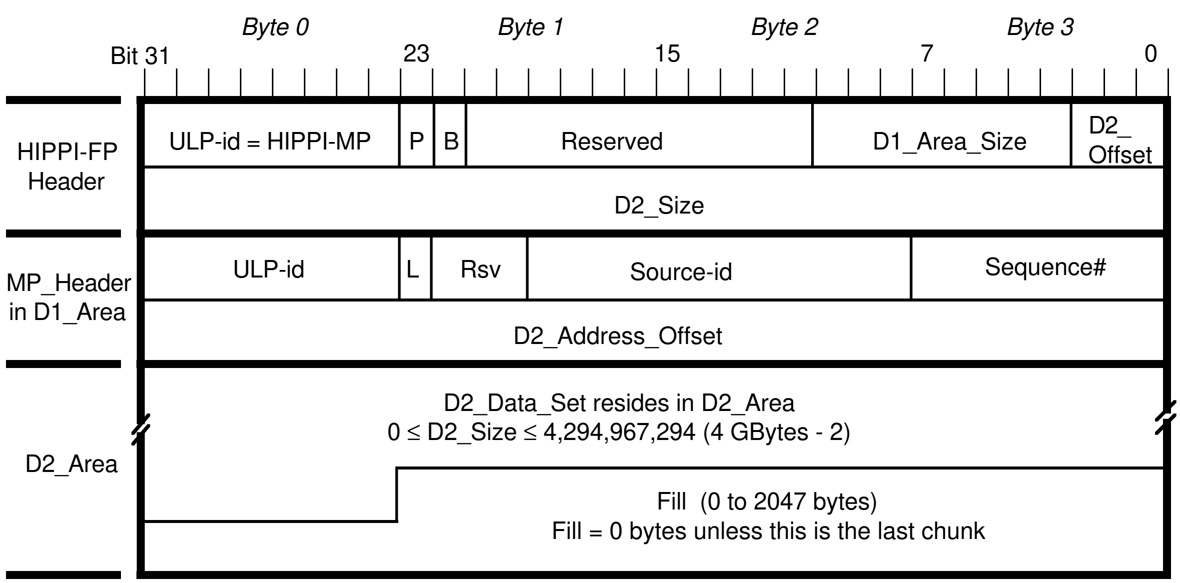

Figure 7 – Second, or later, HIPPI-MP chunk with HIPPI-FP header