US 20170078367A1

(19) United States
(12) Patent Application Publication
Dress et al.
(10) Pub. No.: US 2017/0078367 A1
(43) Pub. Date: Mar. 16, 2017

(54) **PACKET-FLOW MESSAGE-DISTRIBUTION SYSTEM**

(71) Applicant: **LightFleet Corporation**, Camas, WA (US)

(72) Inventors: **William Dress**, Camas, WA (US); **Aaron LeClaire**, Portland, OR (US)

(21) Appl. No.: **15/175,685**

(22) Filed: **Jun. 7, 2016**

Related U.S. Application Data

(60) Provisional application No. 62/216,999, filed on Sep. 10, 2015, provisional application No. 62/217,001, filed on Sep. 10, 2015, provisional application No. 62/217,003, filed on Sep. 10, 2015, provisional application No. 62/217,004, filed on Sep. 10, 2015, provisional application No. 62/241,112, filed on Oct. 13, 2015.

Publication Classification

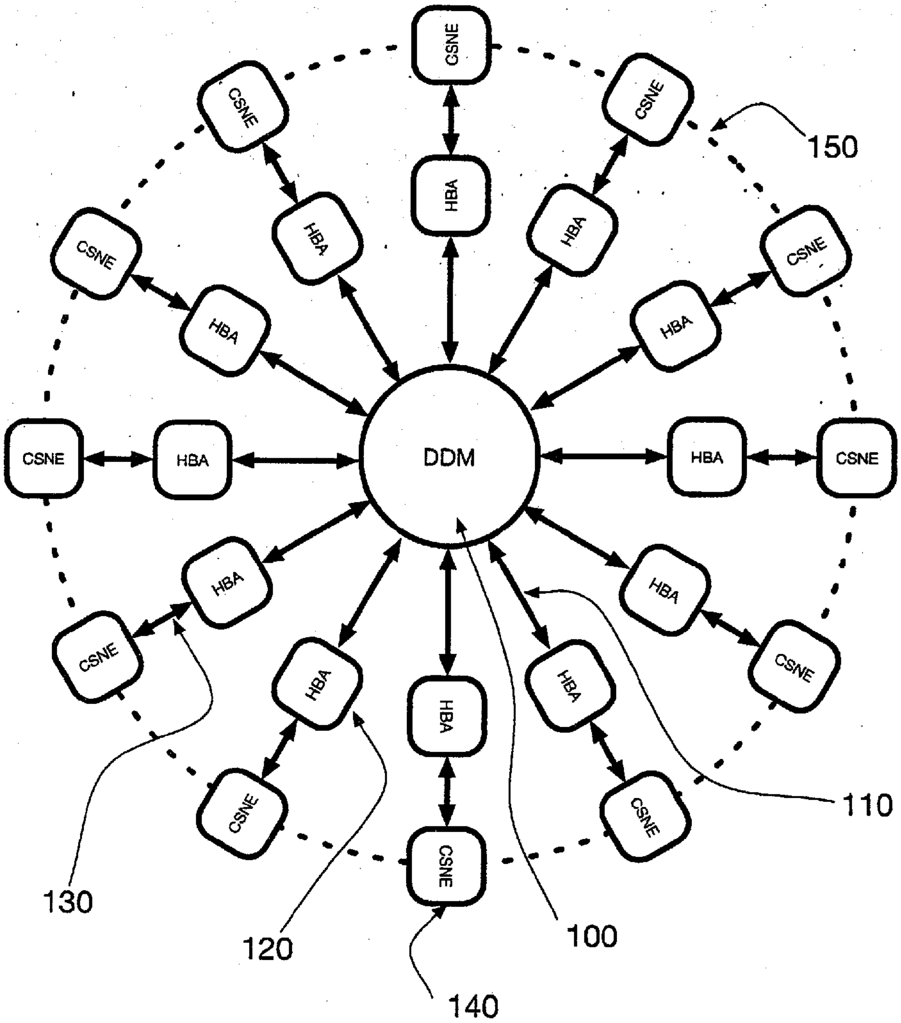
(51) **Int. Cl.**

H04L 29/08	(2006.01)
H04L 12/933	(2006.01)
H04L 12/40	(2006.01)

(52) **U.S. Cl.**
CPC ***H04L 67/10*** (2013.01); ***H04L 12/40163*** (2013.01); ***H04L 49/15*** (2013.01)

(57) **ABSTRACT**

Operating a message distribution can include end-to-end partitioning of message pathways and/or multiple priority levels with interrupt capability.



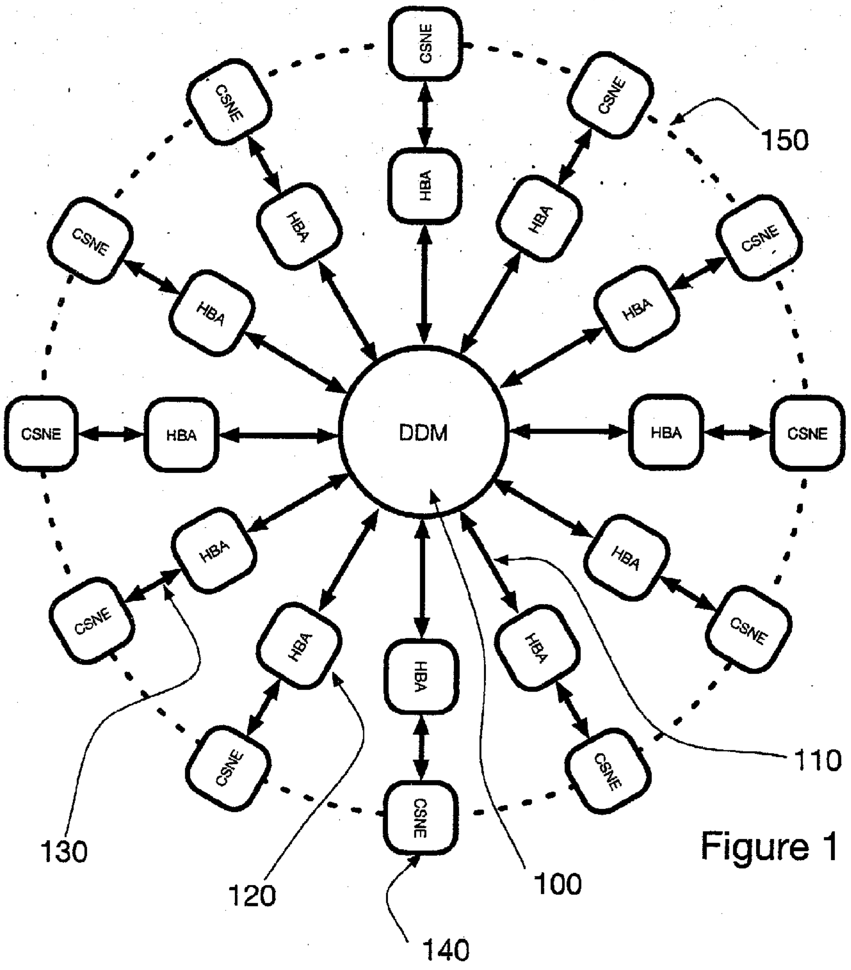


Figure 1

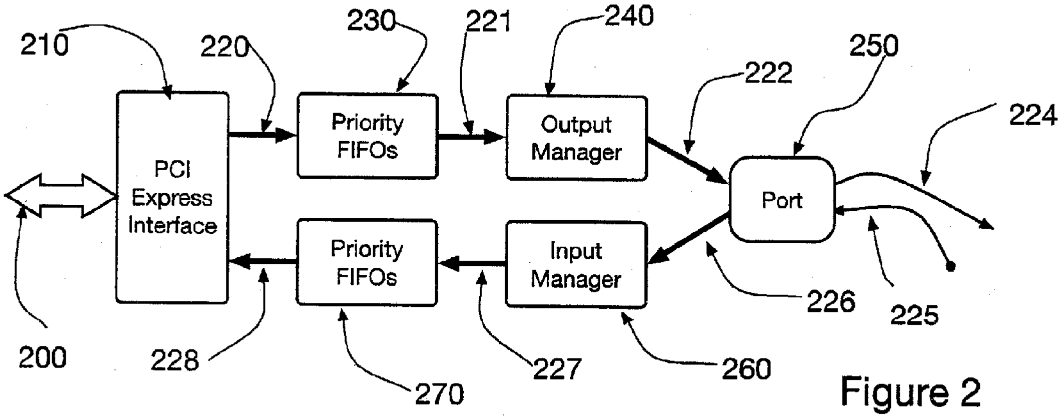


Figure 2

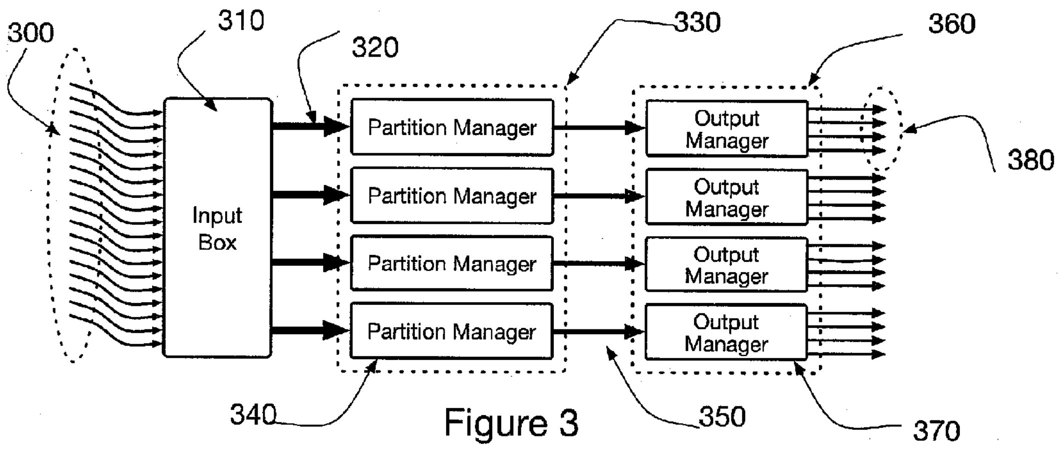


Figure 3

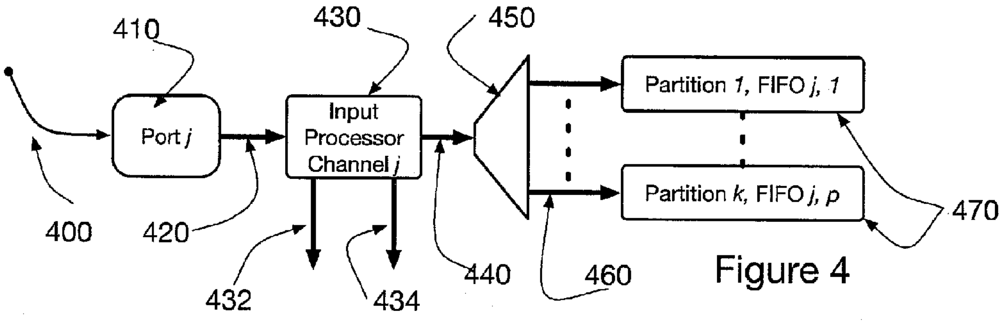

410
430
450
Port *j*
Input Processor Channel *j*
Partition *1*, FIFO *j*, *1*
Partition *k*, FIFO *j*, *p*
470
400
420
432
434
440
460

Figure 4

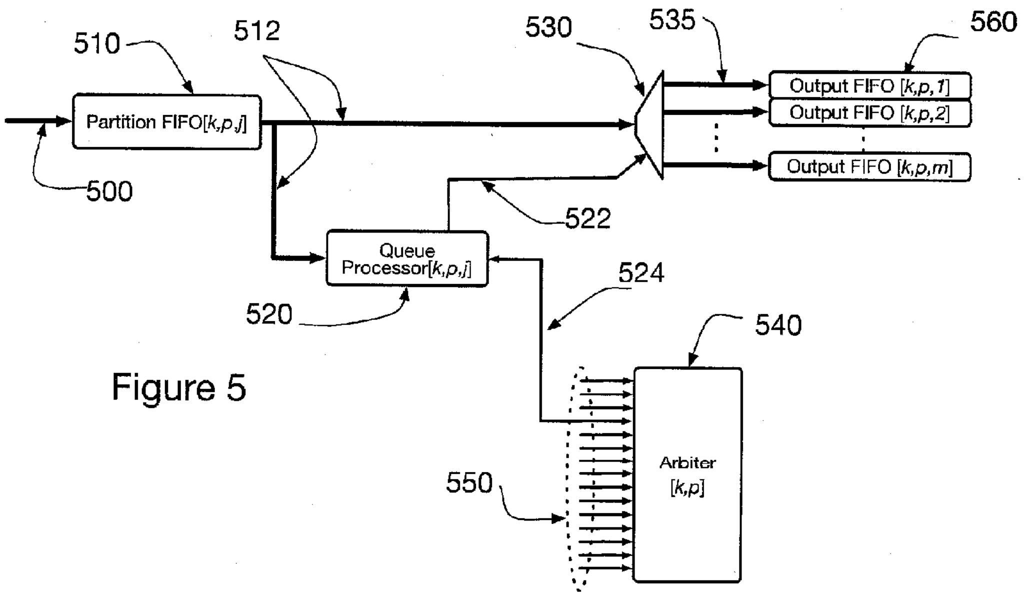

510
512
530
535
560
Partition FIFO[*k*,*p*,*j*]
Output FIFO [*k*,*p*,*1*]
Output FIFO [*k*,*p*,*2*]
Output FIFO [*k*,*p*,*m*]
500
522
Queue Processor[*k*,*p*,*j*]
524
520
540
Arbiter [*k*,*p*]
550

Figure 5

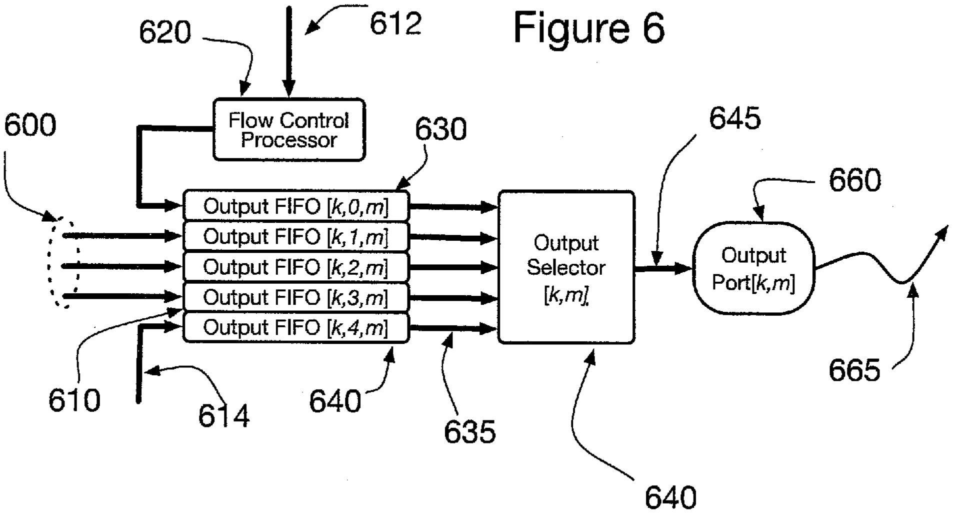


Figure 6

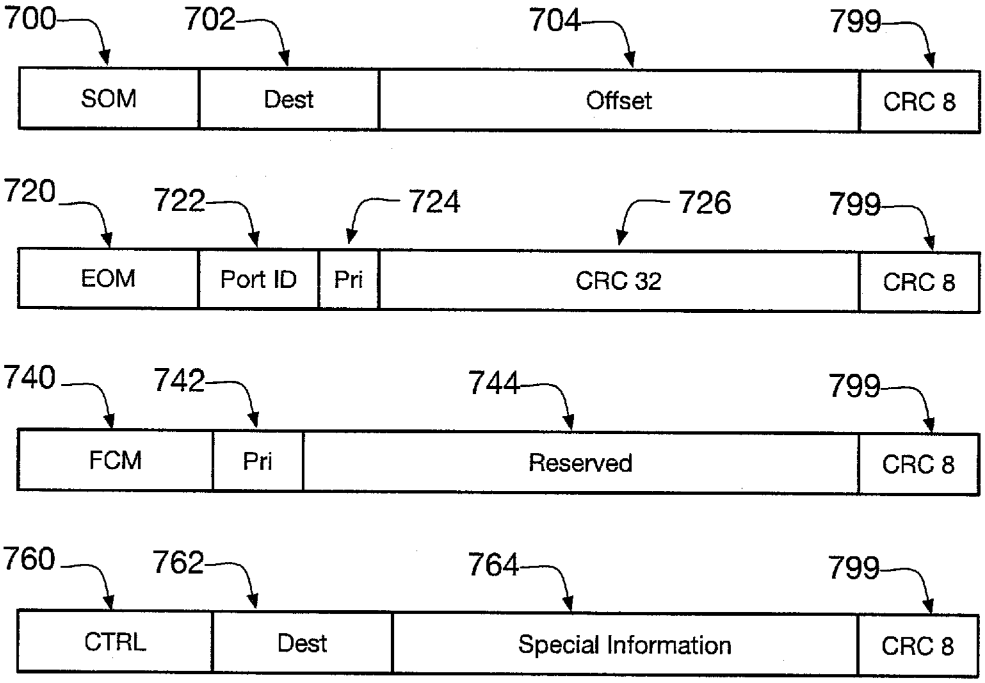


Figure 7

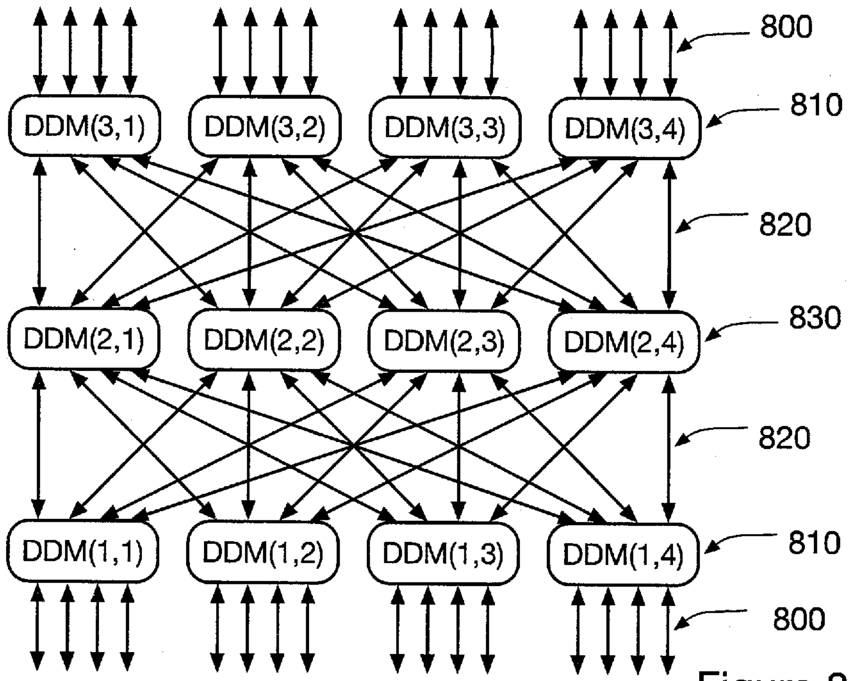

800
810
DDM(3,1)
DDM(3,2)
DDM(3,3)
DDM(3,4)
820
DDM(2,1)
DDM(2,2)
DDM(2,3)
DDM(2,4)
830
820
DDM(1,1)
DDM(1,2)
DDM(1,3)
DDM(1,4)
810
800

Figure 8

PACKET-FLOW MESSAGE-DISTRIBUTION SYSTEM

CROSS-REFERENCE TO RELATED APPLICATION

[0001] Referring to the application data sheet filed herewith, this application claims a benefit of priority under 35 U.S.C. 119(e) from co-pending, commonly-assigned provisional patent applications U.S. Ser. No. 62/216,999, filed Sep. 10, 2015, U.S. Ser. No. 62/217,001, filed Sep. 10, 2015, U.S. Ser. No. 62/217,003, filed Sep. 10, 2015, U.S. Ser. No. 62/217,004, filed Sep. 10, 2015 and U.S. Ser. No. 62/241,112, filed Oct. 13, 2015, the entire contents of all of which are hereby expressly incorporated herein by reference for all purposes.

BACKGROUND

[0002] The invention relates generally to the field of multicomputer interconnects. An interconnect for a multicomputer system or cluster of cooperating hosts, servers, or other processing or storage elements is meant to connect separate and self-contained devices each with its own complement of control, local memory, and other resources. Such clusters are found in cloud storage and processing systems, enterprise computing, large database systems, as well as in high-performance computing.

[0003] A typical installation may include of a number of "tight" clusters which are interconnected by a single dedicated interconnect that may have auxiliary links to routers, the Internet, or directly to other interconnects. Multiple tight clusters may also be interconnected by a homogeneous fabric of like interconnects or by a heterogeneous hierarchy of interconnects of increasing complexity. The latter approach is fast giving way to the former which is based on regular topologies such as the Clos network or the high-dimensional hypercube network in homogeneous interconnect modules.

SUMMARY

[0004] There is a need for the following embodiments of the present disclosure. Of course, the present disclosure is not limited to these embodiments.

[0005] According to an embodiment of the present disclosure, a method comprises: operating a message distribution system includes end-to-end partitioning of message pathways. According to another embodiment of the present disclosure, an apparatus comprises: a message distribution system including a data distribution module and at least two host-bus adapters coupled to the data distribution module, wherein the message distribution system includes end-to-end partitioning of message pathways. According to another embodiment of the present disclosure, a method comprises operating a message distribution system includes multiple priority levels with interrupt capability. According to another embodiment of the present disclosure, an apparatus comprises: a message distribution system including a data distribution module and at least two host-bus adapters coupled to the data distribution module, wherein the message distribution system includes multiple priority levels with interrupt capability.

[0006] These, and other, embodiments of the present disclosure will be better appreciated and understood when considered in conjunction with the following description and the accompanying drawings. It should be understood, however, that the following description, while indicating various embodiments of the present disclosure and numerous specific details thereof, is given for the purpose of illustration and does not imply limitation. Many substitutions, modifications, additions and/or rearrangements may be made within the scope of embodiments of the present disclosure, and embodiments of the present disclosure include all such substitutions, modifications, additions and/or rearrangements.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] The drawings accompanying and forming part of this specification are included to depict certain embodiments of the present disclosure. A clearer concept of the embodiments described in this application will be readily apparent by referring to the exemplary, and therefore nonlimiting, embodiments illustrated in the drawings (wherein identical reference numerals (if they occur in more than one view) designate the same elements). The described embodiments may be better understood by reference to one or more of these drawings in combination with the following description presented herein. It should be noted that the features illustrated in the drawings are not necessarily drawn to scale.

[0008] FIG. **1** illustrates a concept of a tightly coupled cluster of computing or storage or network elements.

[0009] FIG. **2** depicts essentials of a HBA (host-bus adapter) needed to support a DDM (data distribution module).

[0010] FIG. **3** shows the several components and their interconnections that define a data distribution module (DDM).

[0011] FIG. **4** lays out internals of the data distribution module's (DDM's) input stage.

[0012] FIG. **5** illustrates the mechanism from the FIFOs (First In First Out buffers) in the interior of the data distribution module (DDM) to the output stage.

[0013] FIG. **6** depicts an output process that allows priority interrupts.

[0014] FIG. **7** defines the control-frame protocol fundamental to the operation and control of the distribution of messages.

[0015] FIG. **8** gives an example of a type of fabric of data distribution modules (DDMs).

DETAILED DESCRIPTION

[0016] Embodiments presented in the present disclosure and the various features and advantageous details thereof are explained more fully with reference to the nonlimiting embodiments that are illustrated in the accompanying drawings and detailed in the following description. Descriptions of well known materials, techniques, components and equipment are omitted so as not to unnecessarily obscure the embodiments of the present disclosure in detail. It should be understood, however, that the detailed description and the specific examples are given by way of illustration only and not by way of limitation. Various substitutions, modifications, additions and/or rearrangements within the scope of the underlying inventive concept will become apparent to those skilled in the art from this disclosure.

[0017] The invention presented in this disclosure includes one or more data-distribution modules (DDMs) that mediate message transfers within a cluster or set of host processing stations via host-bus adapters (HBAs). The same internal protocol between host-bus adapters (HBAs) mediated by the DDM may be used to communicate between DDMs in a fabric version of the invention, allowing large numbers of hosts to be efficiently interconnected. The resulting message-distribution system (MDS) is a collection of interconnected HBAs and DDMs and the subject of this disclosure.

[0018] An earlier version of the DDM has been described in several issued patents and in several pending applications. The purpose of this disclosure is to present a system concept for which the previously disclosed DDM (variously termed DBOI (Direct Broadcast Optical Interconnect) or broadcast interconnect when based on U.S. Pat. No. 7,450,857 (LIGHT1110-3); U.S. Pat. No. 7,630,648 (LIGHT1110-4); U.S. Pat. No. 8,081,876 (LIGHT1110-5); U.S. Pat. No. 7,970,279 (LIGHT1160); U.S. Pat. No. 8,909,047 (LIGHT1200-1) and US application 2014/0314099 (LIGHT1210-1) and/or WO 2013/142742 (LIGHT1210-WO)) plays an integral part. The entire contents of U.S. Pat. Nos. 7,450,857; 7,630,648; 8,081,876; 7,970,279; and 8,909,047, U.S. Ser. No. 13/848,688, filed Mar. 21, 2013 (U.S. Pat. App. Pub. 20140314099, published Oct. 23, 2014) and Patent Cooperation Treaty Publication No. PCT/US2013/142742 are all hereby expressly incorporated by reference herein for all purposes. Additional architectural and operational features (improvements) which cast a DDM in a system setting are described in this disclosure.

[0019] The main feature of the DDM is based on the concept of message or packet flow where a message or data packet is defined as one more data frames preceded by a start-of-message (SOM) frame and terminated by an end-of-message (EOM) frame. The SOM frame establishes paths within the DDM to the exit ports specified by a destination code contained therein (within the SOM frame). Once a path or paths are established, the rest of the message follows with the EOM releasing the path(s) in use.

[0020] The four main distinguishing features (improvements) present at the architectural or fundamental level of the design of this disclosure are that of the (1) fast-priority message (FPM), (2) the use of multiple priority levels, (3) "end-to-end" partitioning of the message pathways, and/or (4) support for cooperative working groups. The priority of a message is defined by the type of SOM introducing the message into the HBA. This priority is maintained through the DDM to the receiving HBA and then to the receiving host. The basic FPM concept was presented in earlier disclosures, but the "end-to-end" priority channels (partitioning of the message pathways), multiple priority levels with a highest priority channel reserved specifically for FPMs and a channel reserved specifically for maintenance functions and support for cooperative working groups are particular improvements to the MDS design disclosed here. In a preferred embodiment, the management priority is just below the FPM priority to ensure that rarely used management messages get through in a timely fashion. In this embodiment, the packets or normal message traffic has a priority below the management priority.

[0021] All system traffic between components takes place on the same physical layer. That is, in a 48-way system, for example, where the DDM has 48 bidirectional connections to 48 different HBAs, all traffic, whether message traffic or control traffic (FPMs and maintenance messages) travels over these 48 input and 48 output wires (or cables, or fibers). The traffic type is distinguishable only by a type code and not by the particular connection it occupies. (The concept supports any number of inputs and outputs; n=48 in an n-way interconnect is used as an example in the preferred embodiment.)

[0022] Multiple priorities including three or more levels of priority allow flexibility by giving applications the ability to prioritize in more detail certain messages. One example would be to allow efficient and timely periodic memory updates for a group of set of groups by reserving the P1 priority channel for such updates.

[0023] Internally, the DDM is partitioned into one or more sections. Multiple partitions increase the probability of a message being delivered in case of a busy internal path or FIFO carrying a prior message to the same destination. The fundamental result of such partitions is to ensure that the DDM is strictly nonblocking in that a path to a free host whose HBA is not presently receiving a transmission may be connected to any host desiring to send to that free host. That is, the Partition FIFOs do not experience head-of-line blocking such as to prevent a free host from receiving messages. In the MDS, such nonblocking is accomplished without the use of virtual output queues, which are required in switches to ensure nonblocking operation.

[0024] Flow control is a "back pressure" from any host, HBA, or internal FIFO that is nearing capacity and is based on the FPM. A flow-control FPM (FCM) can be issued as an exception by any component along a message path and is sent directly to the DDM or HBA responsible for the exception, bypassing any message in transit so that it arrives in a timely fashion. It is preferred that such a bypass (interrupt) be graceful and not require at its completion the resending of a portion of the interrupted message in transit that has already been sent. Flow control conveys priority information allowing a higher priority to be transmitted even if a lower priority channel (from or to the same HBA) is halted.

[0025] A cooperative working group is a collection of tasks that are distributed among one or more hosts and coordinated by the MDS. A member of a group may only transmit messages to other group members although, in general, group membership can be shared. Many applications, however, require strict separation between groups both for security purposes and to reduce port contention wherein multiple sources require simultaneous access to the same exit port or HBA. The DDM contains a subscription table that is accessed by the group index and specifies to which ports a given message must gain access. The group concept and subscription table provide the low-level support for multicast.

[0026] An internal protocol, similar in structure to an Ethernet packet but with less overhead and greater overall flexibility and message security, is used for the paths HBA to DDM to HBA. Ethernet messages may, of course, be sent to and from the hosts for communicating with entities outside the cluster.

[0027] Those familiar with the art of digital signal transmission and manipulation will understand that there are other ways to effect the transmission from a host message, no matter how presented, to an HBA device that results in a serial data stream suitable for transmission on an optical fiber or other transmission structure to the distribution device, whether switch or DDM.

DETAILED DESCRIPTION

[0028] FIG. **1** illustrates the MDS as a collection of hosts, labeled CSNE **140** for computing, storage, or network element attached to host-bus adapters or HBAs **120** by data lines **130** such as PCI-express **130**. The HBAs send one or more serial data streams via optical fibers (in the preferred embodiment) **110** to the data-distribution module DDM **100**. Dashed line **150** represents any desired number of CSNE-HBA pairs connected to the central distribution hub **100**.

[0029] Since a CSNE **140** (computer, storage, or network element) can represent any of a number of device types, which is not of concern for this disclosure. HBA **120**, however, must be compatible with the DDM and present data by a protocol that is understood and operated upon by DDM **100**. FIG. **2** illustrates a compatible HBA, showing its connection **200** from a host **140** and the interface **210** including, in the preferred embodiment, a multiple-lane, high-speed PCI Express interface along with formatting process and buffer area to convert and order the PCI data streams into parallel lanes compatible with the transmission format chosen. In the preferred embodiment, the transmission of information between system elements, namely between HBA and DDM and, in the fabric version, between DDMs, takes place over high-speed serial links using the industry-standard 64b66b or 64b67b Interlaken line code whereby a frame of 64 bits is encoded as a 66 or 67 bit serial data stream. The preprocessing of the message results in internal pathways in the HBA of 64 bits (in the preferred embodiment but not so limited by the invention to any particular path width) allowing a manageable internal clock rate much lower than the bit rate of the serial connection.

[0030] These 64-bit frames parsed from the input **200**, along with destination and priority information from the host, are stored in the priority FIFOs **230** via the connections **220**. There is one such FIFO **230** for each priority where the number of priorities is a system-design parameter. In a preferred embodiment there are five levels (priorities) including a maintenance level and a highest fast priority message level. The Output Manager **240** has access to each FIFO along with the requested destination via connections **221**. Manager **240** prepares a SOM containing an identification (ID) of the destination group along with an offset into the destination host's memory so that the receiving HBA may store the entire received data by a direct memory access (DMA) action. The result of this operation is to prepare the message for a remote direct memory access (RDMA) from sender to receiver across the cluster. Manager **240** additionally computes a 32-bit cyclic-redundancy check (CRC) using standard algorithms, and places the result in the EOM which is appended to the message as the final frame. The entire vector of frames is presented sequentially to the Bidirectional Port **250** for conversion into an optical, serial data stream which is coupled to output fiber **224**.

[0031] Output Manager **240** operates according to the same prescription as the Output Manager in the DDM (see below for a comprehensive description). On the input side, a serialized optical signal is received at the Bidirectional Port **250** via fiber connection **225** where it is converted to a parallel electrical bit stream and presented to the Input Manager **260** via connections **226**. Input Manager **260** identifies the incoming data stream as to its priority and sends it to the appropriate priority FIFO **270** via connections **227**. A mechanism is provided in Input Manager **260** that allows a higher-priority message to interrupt a lower-priority one by being queued into the FIFO of the correct priority. The priority code in the SOM is decoded to effect this switching, while an EOM switches back to loading the interrupted FIFO. The FIFOs **270** are sent to the PCI Express Interface **210** via connections **228** as the message in a FIFO is completed by reception of the EOM. The interface **210** then formats the message and sends it to the host over connection **200**. Input Manager **260** operates according to the same prescription as the Input Manager in the DDM (see below for a comprehensive description).

[0032] FIG. **3** illustrates the data flow to, through, and from the DDM. Output lines **224** in the HBA are connected to the DDM's input stage, InputBox **310** via fiber connections **300**. InputBox **310** extracts and decodes the destination information from the SOM to select the partition(s) and priority channels for queuing the incoming message in MiddleBox **330**. The message is then sent along connections **320** to the chosen Partition Managers **340** in MiddleBox **330**, where it is either queued, if a delay is required, or it is immediately passed on to a Partition Manager **370** in OutputBox **360** via connections **350**. The partition concept extends from MiddleBox **330** through OutputBox **360**. The priority structure is maintained throughout these partitions and priority interrupts, where a higher-priority message from an HBA can interrupt a lower-priority one by a simple path change made in InputBox **310**, allowing the stream of incoming frames on a connection **300** to switch to the higher-priority queue within a Partition Manager **340**.

[0033] Partition Manager **340** makes a connection to Output Manager **370** in the same partition when the SOM is cleared for transmission to the OutputBox **360**. Output Manager **370** then queues the message frames in a FIFO corresponding to the priority and exit port specified by the SOM's destination code. The message is then passed directly to the output port in the Partition Manager **370** where it is converted to a serial optical signal and sent along output fiber **380**, or it is held in Output Manager **370** should the port be busy with another transmission.

[0034] In greater detail, suppose HBA j sends a message prefixed by a SOM containing the destination, priority, and an offset. This message will appear on input line **400** and enter the InputBox **310** on Port **410** where it is converted to an electrical signal, deserialized, and sent to the Input Processor Channel **430** via the parallel data lines **420**. The Input Processor Channel **430** extracts the destination and priority from the corresponding fields in the SOM. The destination is an index into a subscription table maintained in the InputBox **310** by the aforementioned maintenance messages. The InputProcessor **430** retrieves the exit code from the subscription table and decodes this information to obtain (1) the list of partitions that are to receive copies of the message and (2) the exit indices in the form of an exit map for each of the receiving partitions.

[0035] The exit map for each partition is sent to Distributor **450** via connections **440** which contains a separate path for each of the partitions. Distributor **450** sends a copy of the SOM accompanied by the relevant exit map to the specified Partition FIFO **470** via connection **460** and sets a mux in Distributor **450** to connect the message channel to the indicated Partition FIFO **470**.

[0036] A consequence of the selection process effected by Input Processor **430** and implemented by Distributor **450** is to send the incoming message frames from channel j onto the selected lines **460** to be queued into the selected Partition

FIFOs **470** according to the exits **460** decoded from the destination and priority fields carried by the SOM. Note that a SOM may specify a single priority and multiple partitions. All subpartitions k,p belonging to input index j indicated in FIG. **4** will receive a copy of the message so directed.

[0037] Connection **432** transfers any FCM frame transmitted by HBA j directly to the OutputBox **360**, bypassing the MiddleBox **330** to maintain proper flow control from exit port j to prevent the input FIFO in Input Manager **260** from overflowing. Similarly, connection **434** transmits any response requested by a maintenance message received by the input stage from HBA j for queuing in a maintenance output FIFO (see below).

[0038] Considering the path choices **320** in FIG. **3** together with the path choices **460** in FIG. **4** that are available to any input port, it becomes clear that a message on any input **300** of FIG. **3** can be sent along a path to any of several Partition FIFOs **470** that are specified by the destination and priority fields.

[0039] FIG. **5** illustrates the data flow from the MiddleBox **330** to the OutputBox **360** as shown in FIG. **3** and subsumed in Partition Manager **340**. Partition FIFO[k,p,j] **510** where k specifies the partition, p the priority, and j the input port, receives frames introduced by a SOM on input line **500** as determined by Input Processor **430**. Each Partition FIFO **510** has an associated Queue Processor **520** (indexed by the same {k,p,j} as the FIFO) that has access to the head or output frame residing in that FIFO via connection **512**. The Queue Processor **520** may then determine if a frame is present or not, if the frame is a SOM or EOM, or simply a data frame.

[0040] When a SOM is identified as presenting a message with a set of exits {m} (for the subpartition k,p), this information is passed to corresponding Arbiter[k,p] **540** via connections **524**. The dotted line **550** represents connections to and from the remaining Queue Processors [k,p,{j}] where {j} represents a collection of indices representing all input ports in the DDM.

[0041] Arbiter[k,p] **540** compares the requested set of exits {m} with an internal exit map that maintains the state of the set of Muxes **530** corresponding to the entire set of Output FIFOs. If the bits in this map corresponding to the set {m} are not set, a release is granted by sending notification back to the Queue Processor **520**, which sets the state of Mux **530** via lines **522** so that the SOM may be transferred to the set of Output FIFOs[k,p, {m}] **560** via the selected connections **535**. The set of locations corresponding to {m} are then set in the map to prevent conflicting messages from being released into the Output FIFOs **560**.

[0042] If the set {m} of locations in the bit map are not all clear, the request is not granted and the SOM must wait in Partition FIFO **510** until such time that any conflicting messages have completed their transit from the MiddleBox **330** to the OutputBox **360**.

[0043] The state machine in each Queue Processor[k,p,j] **540** periodically queries Arbiter **540** as long as a SOM remains at the front of the Partition FIFO[k,p,j] **510**, ensuring that any pending message will be released as soon as the map bits {m} are clear.

[0044] Once the SOM has been released and transferred to the specified Output FIFOs **560**, the rest of the message follows by a handshake process controlled, for example, by read- and write-enable flags in the transmitting and receiving FIFOs. These flags reflect the current state of their respective FIFOs in the preferred embodiment, with the read enable set when there is a frame present in Partition FIFO **510** and the write-enable set in Output FIFO **560** as long as there is room in that FIFO for the next frame to be transferred. Note that all Output FIFOs **560** in the released set {m} must be write enabled for transfer to take place.

[0045] The transfer continues, frame-by-frame, until an EOM is transmitted. This event, detected by the Queue Processor **520**, clears the bits {m} in the exit-port map in Arbiter **540**, thus allowing any following message access to those Output FIFOs **560** that received the EOM.

[0046] FIG. **6** illustrates the role of the Output Manager **370** in OutputBox **360** as a process managing message traffic from all Output FIFOs [k, {p}, m] **560** to Output Port[k,m] **660** in partition k for the specified output m in that partition. {p} is the set of message priorities supported by the MDS. This manager for partition k serves only exit m by mediating the traffic from all priorities from 0 to P+1, where P is the number of message priorities supported by the MDS. There is one such manager for each of the n output ports in the MDS. FIG. **6** depicts the preferred embodiment with P=3, resulting in a total of 5 Output FIFOs **630** in each of the K partitions.

[0047] Inputs **600** from the set of MiddleBox Partition Managers as shown in FIG. **5** send message frames to the Output FIFOs **610** from each message priority as mediated by the set of Arbiters **540** in partition k for any or all of the P priorities. Input **614** allows any maintenance message prepared by Input Processor **430** on input channel j to be queued in Output FIFO[k,4,m] via connection **434** where m=j mod M so the response is sent to the requesting HBA j. Likewise, input **612** contains information regarding any of the Partition FIFOs[k,p,j] **470** that generate a flow control exception. Input **612** also transmits any FCM sent by an HBA and processed in Input Processor Channel[j] **430** via connection **432**. Both of these notifications are processed by the Flow Control Processor **620** which either formats an FCM for delivery to HBA j in the case of a notification from a Partition FIFO **470** or controls the flow of frames to the Output Port[k,m] in the case of a notification from the Input Processor Channel[j].

[0048] If one or more of the Output FIFOs[k, {p}, m] contain at least one frame, Output Selector[k,m] selects the appropriate Output FIFO **610**, **630**, or **640**, depending upon priority, for transmission to Output Port [k,m] **660** via connection **645**. Output Port **660** then serializes the frame of 64 bits (in the preferred embodiment), converts the electrical signal to a serial optical stream, and injects the bit stream onto the optical fiber **665** leading to HBA j. The details of this conversion and transmission process are well known to practitioners of the art of serial digital data transmission via optical fibers.

Internal Protocol

[0049] Externally (messages flowing into or out of the system of hosts, HBAs, and DDMs) one of the common protocols, such as Ethernet, may be used. Internally, messages are wrapped in the SOM and EOM for efficient and reliable transfer between hosts. In normal operation, the host sends a data packet in the form of destination, priority, message length and a sequence of bytes containing the message payload to its HBA. The payload contains arbitrary information and may include other communication protocols for routing outside the MDS. The function of the HBA is to decompose these data into frames (of 64 bits in the preferred embodiment), prefix the sequence of frames with the SOM control frame which contains information as to the packet's destination, priority, and offset into the receiving host's memory for the RDMA function. The HBA also prepares and appends the EOM control frame which contains a 32-bit CRC error-detecting code and a source designation identifying the transmitting HBA.

[0050] FIG. **7** illustrates the MDS' transmission and control protocol in the preferred embodiment. Variations on this basic theme in keeping with the essential functions of group identification for multicast, DMA access into the destination memory, and reliability are, of course, possible. Each of these control frames is 64 bits long with the first section of the frame reserved for the control type (SOM **700** in the case of the SOM header, EOM **720** in the case of the EOM tail, FCM **740** for FCMs, and CTRL **760** for other types of control frames). These frame-type identifiers (**700**, **720**, **740**, and **760**) are decoded in the InputBox **310** and result in different actions in the Input Box **310** as discussed above. Each control frame has an error code **799** occupying the final byte of the frame. This code may be used for error detection or error correction, depending on the algorithm used to compute the code; both operations are possible, but generally require different state machines in the HBAs and a different error-check in during the input-stage process.

[0051] The control-type identifier takes from 6 to 12 bits depending on the serial interface used to transmit and receive the information. In the preferred embodiment, these identifiers require 4 bits allowing for 16 different control types. The prefix bits **66**:**64** in the 64b66b encoding identify any frame as to a control or a data frame (or an error) according to the standard protocol for this encoding. Dest field **702** contains a group identifier of 12 bits in the preferred embodiment, allowing a full 2048 groups to be identified in the DDM (whose subscription table, in this case, would contain 2048 entries). Offset field **704** contains an offset into the memory of the destination offset from a base address, which specifies the start of the memory reserved to the specific group receiving the message.

[0052] The EOM is constructed in the same manner with EOM type **720** including of 4 bits in the preferred embodiment. The field Port ID **722** contains a numerical identification of the HBA sending the message. This identification may be used in the receiving HBA or host to prepare an acknowledgement (ACK) or negative acknowledgement (NAK) to demand a resend should the MDS operate under a specific error-recovery protocol. The EOM frame also optionally contains a Priority **724** field that may be used for a variety of security and recovery purposes (not discussed in this disclosure). The main function of the EOM frame is to convey the CRC 32 field **726** so that the integrity of the data frames lying between the SOM and EOM may be verified. This check is made in the receiving HBA, where, in case of an error, the entire message may be dropped or reported to the host so that a recovery or retransmission-request process may be initiated.

[0053] Each CRC 8 code **799** contains a check sum or error-correcting code as describe above to protect the integrity of the first 56 bits of the control frame. In this way, essential control frames, such as FCMs for flow control, may be acknowledged via an ACK control frame (CTRL **760**). The code **799** for the SOM frame causes the frame and entire message to be dropped in case of an error. This prevents a catastrophic system error where the wrong host could have its memory overwritten or the wrong memory location in a valid host could be overwritten. The since the error code **799** is checked in the Input Processor **410**, the transmitting HBA may be notified immediately by an error FPM control code **760** passed directly to the correct Output FIFO **630**. In this case, the transmitting host would have the option to resend the message. If an error in the SOM is detected in the receiving HBA, the receiving host would have the option to request a resend by extracting the identification of the sending host from the EOM.

[0054] The FCM control frame contains a Priority field **742** that contains a bit map of the combined state of the Partition FIFOs[k,p,j] receiving information from HBA j. This allows HBA j to either halt or restart a transmission from any of the Priority FIFOs **230**.

[0055] An error detected on code **799** in an EOM transmission to the DDM would initiate much the same action as described in the previous paragraph. A EOM error detected in the receiving EOM may require a more general type of error recovery. However, if code **799** were an ECC (error-correcting code) such as Hamming (63,57) code, most all errors in the control frames could be corrected without the need for error-recovery in the host. Of course, this depends on the random bit-error rate experienced by the physical transport layer (in the preferred embodiment, this rate is experimentally 10^{-16}, meaning that the frequency of occurrence of an uncorrectable error in a control frame would be vanishingly small).

[0056] It is seen, by the above discussion, that the control frames presented in FIG. **7** allow a versatile and secure operation of the MDS. Data integrity is protected end-to-end, from transmitting HBA, through the DDM and over the fiber connections, to the receiving HBA. Multicast messages (see the discussion on groups below) are supported at both the internal protocol level and by the processes inherent in the DDM from the InputBox **310**, through the MiddleBox **330**, to the OutputBox **360**.

Groups and Message Destination

[0057] The above detailed description of the drawings explains the relationship between the message header and the self-routing process that is inherent in the SOM's various fields and the mechanism of message management illustrated in FIGS. **3**, **4**, **5**, and **6**. A destination for a message includes two parts: the group ID and the offset. The group ID, in the preferred embodiment, serves as an index into a table of group subscriptions, said table being located in InputBox **310** for efficient access to the process that directs any message to the correct partitions and destinations.

[0058] Groups are defined at the application level and their table entries are communicated to the DDM by maintenance messages generated by a group manager process in a particular host. Such messages are directed the DDM itself and are not passed through to other hosts.

Fast-Priority Messages

[0059] Fast priority messages, such as the FCMs, allow the MDS to respond quickly and efficiently to a variety of situations, from FIFO management to reporting component failures in a timely fashion. Such messages are quick to transmit since each includes a frame of 64 bits that is transmitted at wire speed to any location in the cluster. The brevity and speed of these control messages enables a rapid system response to changing conditions as well as supports a graceful degradation of the system when hosts or HBAs are lost.

DDM Fabric

[0060] Fabric topologies based on interconnecting identical DDMs are the subject of patent application WO 2013/142742 (LIGHT1210-WO). The full impact of the self-routing packet is not fully evident until a fabric of multiple interconnected MDS modules are deployed. At each step across a fabric, the SOM opens a path in the most direct and efficient manner available to it at the moment it enters a fabric module. Path segment availability information is continually updated throughout the fabric by the back-pressure concept based on the FCMs as discussed above. The advantage is that a global view of the state of message traffic is not required; indeed, a global supervisory control function based on such global knowledge, with its attendant reliance on specialized spanning-tree algorithms, is known to be unreliable and certainly adds to latency across the fabric. These issues are simply not present in a DDM fabric where each SOM effects path decisions as the message traverses the fabric.

[0061] A DDM fabric includes multiple identical DDMs interconnected according in a specific topological arrangement such as a Clos or hypercube topology, to give two examples. An example of the Clos topology is shown in FIG. **8** which includes three layers of DDMs each with 8 bidirectional ports, is labeled by its position in the two-dimensional array with the first index referring to the row and the second to the column position in the array. Connections **800** are to the HBAs in the top and bottom rows while connections **820** are between row **1** and row **2** HBAs as well as between row **2** and row **3** HBAs such that each HBA in a row is connected to every HBA in an adjacent row. Row **1** and row **3**, labeled **810** comprise the outer rows while row **2**, labeled **830** is the inner row. A Clos fabric based on modules with n ports contains 3/2 n fabric modules (interconnects), $2(n/2)^2$ connections to HBAs, and $n^2/2$ interior connections between rows. The maximum number of hops to reach any HBA from any other, also known as the fabric diameter, is 3 for the Clos network. This topology is significantly more efficient in number of hops and hardware required than a tree structure interconnecting the same number of HBAs.

[0062] For such a fabric to support the inherent advantages of the self-routing messages described above, the subscription tables for multicast routing must be tailored to the specific topology. In the case of the Clos network of FIG. **8**, each DDM requires a subscription table based on its location (or index pair) within the fabric so that the destination carried by an entering SOM is mapped to the correct output ports. The details depend on the assignment of HBA indices as well. The computation of these tables is based on a simple algorithm and the tables may be updated, with group membership changing as the application demands. Each subscription table is maintained by the process described above for maintenance messages.

[0063] Embodiments of this disclosure can include the use of a simple subscription table containing an exit map for each defined group. In embodiments of this disclosure, traffic flow through a fabric of identical DDMs depends the contents of the subscription table in each DDM.

[0064] Embodiments of this disclosure can include the fast-priority message that uses the same data links between MDS components as other messages. Most interconnect systems are based on Ethernet or Infiniband that both require much longer control messages than the fast-priority message 64 bits and/or are made over separate physical connections. Agile flow control based on the fast-priority message does not require additional connections or access to a control plane or supervisory traffic manager.

DEFINITIONS

[0065] The phrase end-to-end partitioning of message pathways is intended to mean partitioning of the message pathways from a CSME (computing, storage, or network element) to another CSME, for instance a priority channel from a computing element through a host-bus adapter through a data distribution module through another data distribution module then through another host-bus adapter and then to a storage element. The phrase multiple priority levels is intended to mean three or more priority levels, for instance five priority levels including a highest priority channel reserved specifically for fast priority messages and a channel reserved specifically for maintenance functions. The terms program and software and/or the phrases program elements, computer program and computer software are intended to mean a sequence of instructions designed for execution on a computer system (e.g., a program and/or computer program, may include a subroutine, a function, a procedure, an object method, an object implementation, an executable application, an applet, a servlet, a source code, an object code, a shared library/dynamic load library and/or other sequence of instructions designed for execution on a computer or computer system).

[0066] The term uniformly is intended to mean unvarying or deviate very little from a given and/or expected value (e.g, within 10% of). The term substantially is intended to mean largely but not necessarily wholly that which is specified. The term approximately is intended to mean at least close to a given value (e.g., within 10% of). The term generally is intended to mean at least approaching a given state. The term coupled is intended to mean connected, although not necessarily directly, and not necessarily mechanically.

[0067] The terms first or one, and the phrases at least a first or at least one, are intended to mean the singular or the plural unless it is clear from the intrinsic text of this document that it is meant otherwise. The terms second or another, and the phrases at least a second or at least another, are intended to mean the singular or the plural unless it is clear from the intrinsic text of this document that it is meant otherwise. Unless expressly stated to the contrary in the intrinsic text of this document, the term or is intended to mean an inclusive or and not an exclusive or. Specifically, a condition A or B is satisfied by any one of the following: A is true (or present) and B is false (or not present), A is false (or not present) and B is true (or present), and both A and B are true (or present). The terms a and/or an are employed for grammatical style and merely for convenience.

[0068] The term plurality is intended to mean two or more than two. The term any is intended to mean all applicable members of a set or at least a subset of all applicable members of the set. The phrase any integer derivable therein is intended to mean an integer between the corresponding numbers recited in the specification. The phrase any range derivable therein is intended to mean any range within such corresponding numbers. The term means, when followed by the term "for" is intended to mean hardware, firmware and/or software for achieving a result. The term step, when followed by the term "for" is intended to mean a (sub) method, (sub)process and/or (sub)routine for achieving the recited result. Unless otherwise defined, all technical and scientific terms used herein have the same meaning as commonly understood by one of ordinary skill in the art to which this present disclosure belongs. In case of conflict, the present specification, including definitions, will control.

[0069] The described embodiments and examples are illustrative only and not intended to be limiting. Although embodiments of the present disclosure can be implemented separately, embodiments of the present disclosure may be integrated into the system(s) with which they are associated. All the embodiments of the present disclosure disclosed herein can be made and used without undue experimentation in light of the disclosure. Embodiments of the present disclosure are not limited by theoretical statements (if any) recited herein. The individual steps of embodiments of the present disclosure need not be performed in the disclosed manner, or combined in the disclosed sequences, but may be performed in any and all manner and/or combined in any and all sequences. The individual components of embodiments of the present disclosure need not be combined in the disclosed configurations, but could be combined in any and all configurations.

[0070] Various substitutions, modifications, additions and/or rearrangements of the features of embodiments of the present disclosure may be made without deviating from the scope of the underlying inventive concept. All the disclosed elements and features of each disclosed embodiment can be combined with, or substituted for, the disclosed elements and features of every other disclosed embodiment except where such elements or features are mutually exclusive. The scope of the underlying inventive concept as defined by the appended claims and their equivalents cover all such substitutions, modifications, additions and/or rearrangements.

[0071] The appended claims are not to be interpreted as including means-plus-function limitations, unless such a limitation is explicitly recited in a given claim using the phrase(s) "means for" or "mechanism for" or "step for". Sub-generic embodiments of this disclosure are delineated by the appended independent claims and their equivalents. Specific embodiments of this disclosure are differentiated by the appended dependent claims and their equivalents.

What is claimed is:

1. A method, comprising operating a message distribution system includes end-to-end partitioning of message pathways.

2. The method of claim **1**, wherein there are at least three priority levels with interrupt capability.

3. The method of claim **1**, wherein there is support for cooperative working groups.

4. The method of claim **1**, further comprising sub-partitioning of message pathways.

5. A non-transitory computer readable media comprising executable programming instructions for performing the method of claim **1**.

6. An apparatus, comprising: a message distribution system including a data distribution module and at least two host-bus adapters coupled to the data distribution module, wherein the message distribution system includes end-to-end partitioning of message pathways.

7. The apparatus of claim **6**, further comprising a computing, storage or networking element coupled to each of the at least two host-bus adapters.

8. The apparatus of claim **6**, further comprising another data distribution module coupled to the data distribution module.

9. A network, comprising the apparatus of claim **6**.

10. An interconnect fabric, comprising the apparatus of claim **6**

11. A method, comprising operating a message distribution system includes multiple priority levels with interrupt capability.

12. The method of claim **11** wherein there is end-to-end partitioning of message pathways.

13. The method of claim **11**, wherein there is support for cooperative working groups.

14. The method of claim **11**, further comprising.

15. A non-transitory computer readable media comprising executable programming instructions for performing the method of claim **11**.

16. An apparatus, comprising: a message distribution system including a data distribution module and at least two host-bus adapters coupled to the data distribution module, wherein the message distribution system includes multiple priority levels with interrupt capability.

17. The apparatus of claim **16**, further comprising a computing, storage or networking element coupled to each of the at least two host-bus adapters.

18. The apparatus of claim **16**, further comprising another data distribution module coupled to the data distribution module.

19. A network, comprising the apparatus of claim **16**.

20. An interconnect fabric, comprising the apparatus of claim **16**

* * * * *