(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
24 September 2015 (24.09.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/141014 A1

(51) International Patent Classification:
G06F 13/36 (2006.01) *G06F 13/14* (2006.01)

(21) International Application Number:
PCT/JP2014/058146

(22) International Filing Date:
18 March 2014 (18.03.2014)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **NEC CORPORATION** [JP/JP]; 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP).

(72) Inventors: **SUN Lei**; c/o NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP). **YOSHIKAWA Takashi**; c/o NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP). **TAKAHASHI Masahiko**; c/o NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP). **SUZUKI Jun**; c/o NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP). **TSUJI Akira**; c/o NEC Corporation, 7-1, Shiba 5-chome, Minato-ku, Tokyo, 1088001 (JP).

(74) Agents: **TANIA Sumio** et al.; 1-9-2, Marunouchi, Chiyoda-ku, Tokyo 1006620 (JP).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

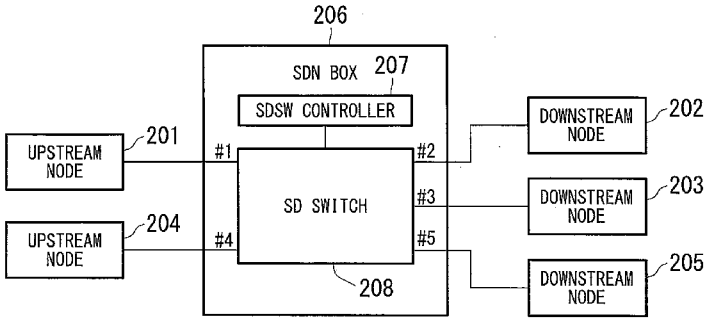
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

(54) Title: METHOD OF CONSTRUCTING SOFTWARE-DEFINED PCI EXPRESS (PCI-E) SWITCH

FIG. 2A



(57) Abstract: The invention generally relates to the technical field of networking and computer architecture, and more specifically relates to approaches of constructing a system which includes a software-defined network (SDN) box and at least one upstream node and one more than downstream node. The nodes may be divided into several groups. In each group, there is a single upstream node and at least one downstream node. All packets between the upstream node and downstream nodes are forwarded by the SDN box. The SDN box acts as a 'PCI-E switch', which interconnects the upstream node to each downstream node and forwards the encapsulated packets (the inner is PCI-E packets). During the whole communication process, the upstream node treats downstream nodes as ordinary local PCI-E devices.

WO 2015/141014 A1

DESCRIPTION

METHOD OF CONSTRUCTING SOFTWARE-DEFINED PCI EXPRESS (PCI-E) SWITCH

TECHNICAL FIELD

The invention generally relates to the technical field of networking and computer architecture, and more specifically relates to approaches of constructing a system which consists of a software-defined network (SDN) box, upstream nodes and downstream nodes.

BACKGROUND ART

PCI-E (Peripheral Component Interconnect-Express) is the third generation high performance I/O bus used to interconnect peripheral devices in computer systems. PCI-E employs the same usage model as previous generation I/O bus called PCI. It supports familiar transactions such as memory read/write, I/O read/write and configuration read/write transactions. Existing OS and device drivers can run in a PCI-E system without any modifications (as to PCI-E in detail, refer to "PCI Express system architecture, by Ravi Budruk, Don Anderson, Tom Shanley. Addison-Wesley Professional, 2004).

The number of PCI-E slots is usually limited because there is not so much space in servers. ExpEther (Express Ethernet; see http: //www.expether.org/.) is proposed to address the above problem. ExpEther extends PCI-E over Ethernet. PCI-E packets are encapsulated to PCI-E-over-Ethernet packets at the side of sender of PCI-E packet; then they are forwarded by Ethernet switches, and when these packets are forwarded to

the destination, they are decapsulated to PCI-E packets (see Japanese Unexamined Patent Application, First Publication No. 2007-219873A).

A certain type of packets is broadcasted to act as a keep-alive message in ExpEther. There is a timeout value associated with it. If timeout expires, there is still no such keep-alive packets received, the upstream node is concluded that it is already out of service. Once failure is detected, the connected downstream nodes should be handed over to another available upstream node as soon as possible.

DISCLOSURE OF INVENTION

In current ExpEther, it is difficult to achieve fast failure detection. Because faster failure detection results in shorter timeout value of keep-alive packets, and shorter timeout value results in more broadcast traffic. More broadcast traffic increases the workload of network devices.

The invention is proposed to solve the above fast failure detection and hand-over problem of ExpEther. It provides a method of constructing a system based on PCI-E, Ethernet and SDN. The proposed system consists of upstream nodes, downstream nodes and a SDN box.

In the proposed system of this invention, there is an encapsulation/decapsulation module on both an upstream node and a downstream node. The upstream node (may consists of CPU, memory, hard disk and various I/O devices) behaves as a computing system, and the downstream node behaves as a PCI-E device, they are connected by the SDN box. The upstream node accesses the downstream node by PCI-E packets. To both upstream node and downstream node, the communication process is as follows. When sending PCI-E packet, it encapsulates the PCI-E packet with a specific packet header, and send it out via the network (e.g. Ethernet but NOT limited to Ethernet).

When receiving the encapsulated packet, it removes the specific packet header, decapsulates it into a PCI-E packet. The SDN box acts as a 'PCI-E switch', which interconnects the upstream node to the downstream node via the specific network and forwards the encapsulated packets.

In the proposed system of this invention, when the upstream node is out of service, a notification will be delivered to SDN box (the SDN box can be implemented by OpenFlow (see https://www.opennetworking.org/sdn-resources/onf-specifications/openflow) but not limited to it. If OpenFlow is used, an *OFPPS_LINK_DOWN* message will be sent from OpenFlow switch to OpenFlow controller when the upstream node is out of service). Moreover, the PCI-E routing table is maintained on the SDN box, the hand-over can also be achieved by modifying the group ID of connected downstream nodes.

BRIEF DESCRIPTION OF THE DRAWINGS

To describe the foregoing and other exemplary purposes, aspects, and advantages, we use the following detailed description of an exemplary embodiment of the invention with reference to the drawings, in which:

FIG. 1 is a block diagram depicting an embodiment of system architecture of a computing system with a SDN box, at least one upstream node and at least one downstream node;

FIG. 2A and FIG. 2B are block diagrams depicting an embodiment of system architecture of one of possible implementations based on a SDN box, which consists of at least one software-defined switch (SD switch) and at least one software-defined switch controller (SDSW controller);

FIG. 3 is a block diagram depicting an embodiment of system architecture of an

upstream node;

FIG. 4 is a block diagram depicting an embodiment of system architecture of a downstream node;

FIG. 5 is a block diagram depicting an embodiment of system architecture of a software-defined (SD) switch;

FIG. 6 is a block diagram depicting an embodiment of system architecture of a software-defined switch controller (SDSW controller);

FIG. 7 is a table depicting an embodiment of a possible packet format of the DEVINFO packet;

FIG. 8 is a sequence diagram of the memory read/write between upstream node, SDN box and downstream node;

FIG. 9 is a flowchart depicting an embodiment of a method of an upstream node or a downstream node how sends out PCI-E packets;

FIG. 10 is a flowchart depicting an embodiment of a method of an upstream node or a downstream node how receives PCI-E packets;

FIG. 11 is a flowchart depicting an embodiment of a method of the SD switch how to handles Ethernet packets;

FIG. 12 is a flowchart depicting an embodiment of a method of the SDSW controller how to set up the master transaction layer packet (TLP) routing table during system initiation and how to communicate with SD switch during the process of packet forwarding;

FIG. 13 is a table depicting an embodiment of a possible data structure of TLP routing table, which is used as slave TLP routing table on the SD switch and master TLP routing table on the SDSW controller, when the underlying network is Ethernet; and

FIG. 14 is a sequence diagram of the failure detection and hand-over between

upstream nodes, SDN box and downstream nodes.

EMBODIMENTS FOR CARRYING OUT THE INVENTION

In the following description, a preferred embodiment of the invention is described with regard to preferred process steps and data structures.

<System components>

FIG. 1 is a block diagram depicting an embodiment of system architecture of a computing system with a software-defined network (SDN) box 106, upstream nodes 101 and 104 and downstream nodes 102, 103 and 105. Each component is connected by network (e.g. Ethernet but not limited to it). Generally there is at least one group in the system. In FIG. 1 there are two groups. One group consists of upstream node 101 and downstream node 102 and downstream node 103. The other group consists of upstream node 104 and downstream node 105. In each group, there is a single upstream node and at least one downstream node. All packets between the upstream node and downstream nodes are forwarded by the SDN box.

<System architecture of a possible implementation>

FIG. 2A and FIG. 2B are block diagrams depicting an embodiment of system architecture of a possible implementation, where the SDN box consists of at least one software-defined switch (SD switch) and at least one software-defined switch controller (SDSW controller).

In FIG. 2A, the SDN box 206 consists of a SD switch 208 and a SDSW controller 207. There is a communication channel between the SD switch 208 and the SDSW controller 207. The upstream nodes and downstream nodes can be divided into two groups, and each group consists of a single upstream node and at least one downstream node. The upstream node 201 and two downstream nodes 202 and 203 are

connected to the first (#1), second (#2) and third (#3) ports of SD switch 208 respectively. The upstream node 204 and the downstream nodes 205 are connected to the fourth (#4) and fifth (#5) ports of SD switch 208 respectively.

In FIG. 2B, the SDN box 206 consists of SD switches 208, 209, 211, and 212 and SDSW controllers 207 and 210. There is a communication channel between SDSW controller 207 and SD switch 208 and 209 respectively. Additionally, there is also a communication channel between the SDSW controller 210 and SD switch 211 and 212 respectively. The upstream nodes and downstream nodes can be divided into two groups, and each group consists of a single upstream node and at least one downstream node. The upstream node 201 and two downstream nodes 202 and 203 belong to one group and the upstream node 204 and the downstream node 205 belong to another group.

In FIG. 2A and FIG. 2B, all SD switches are connected by a specific network, e.g. Ethernet (but not limited to it). The communication between the SDSW controller and SD switches are in a specific protocol, e.g. OpenFlow (but not limited to it).

FIG. 3 is a block diagram depicting an embodiment of system architecture of an upstream node. In general, a computer system consists of CPU, memory, hard disk and various I/O devices. To explain the system architecture more clearly, the rest components are omitted here. An upstream node 301 consists of at least a central processing unit (CPU) 302, an encapsulation/decapsulation (ENCAP/DECAP) module 303 and a network interface card (NIC) 304. CPU 302 is a hardware component that carries out the instructions of a computer program by performing the basic arithmetical, logical, and input/output operations of the system. NIC 304 is a hardware component that connects the upstream node 301 to the network where SDN box is located. The encapsulation/decapsulation module 303 is in charge of encapsulation of PCI-E packets and decapsulation the received encapsulated packets during the process of

communication. CPU 302 and the encapsulation/decapsulation module 303 are connected logically; and the encapsulation/decapsulation module 303 and NIC 304 are connected respectively. In another word, the connecting method is not limited, it may be either hardware method e.g. PCI-E bus protocol (but not limited to it) or any software method.

FIG. 4 is a block diagram depicting an embodiment of system architecture of a downstream node. It may consist of CPU, memory, hard disk and various devices. To explain the system architecture more clearly, the rest components are omitted here. A downstream node 401 consists of at least memory 402, an encapsulation/decapsulation (ENCAP/DECAP) module 403 and a network interface card (NIC) 404. NIC 404 is a hardware component that connects the downstream node 401 to the network where SDN box is located. The encapsulation/decapsulation module 403 is in charge of encapsulation of PCI-E packets and decapsulation the received encapsulated packets during the process of communication. The memory 402 and the encapsulation/decapsulation module 403 are connected; the encapsulation/decapsulation module 403 and NIC 404 are connected logically respectively. In another word, the connecting method is not limited, it may be either hardware method e.g. PCI-E bus protocol (but not limited to it) or any software method.

<Packet processing inside a SDN box>

FIG. 5 is a block diagram depicting an embodiment of system architecture of a software-defined (SD) switch. A SD switch 501 at least consists of a RecvD module 502, a SendD module 503, a SendC (sending to control plane) module 504, a RecvC (reception from control plane) module 509 a decapsulation (DECAP) module 505, an encapsulation (ENCAP) module 506, a packet buffer (PKT BUFFER) module 507 and a slave transaction layer packet (TLP) routing table operation module 508. The RecvD

module 502 is in charge of reception from upstream nodes and downstream nodes. The SendD module 503 is in charge of sending to upstream nodes and downstream nodes. The RecvC module 509 is in charge of reception from SDSW controller. The SendC module 504 is in charge of sending to SDSW controller. The decapsulation module 505 decapsulates the encapsulated packets to PCI-E packets. The encapsulation module 506 encapsulates PCI-E packets. The packet buffer module 507 is in charge of buffering packets. The slave TLP routing table operation module 508 can create and insert a new slave TLP routing table item and support retrieving function of the slave TLP routing table.

FIG. 6 is a block diagram depicting an embodiment of system architecture of a software-defined switch controller (SDSW controller). A SDSW controller at least consists of a RecvSW module 602, a SendSW module 603, a master TLP routing table operation module 604, and a msg-parser module 605. The RecvSW module 602 is in charge of reception from SD switch. The SendSW module 603 is in charge of sending to SD switch. The master TLP routing table operation module 604 can create and insert a new master TLP routing table item and support retrieving function of the master TLP routing table. The msg-parser module 605 extracts info which includes: 1) Node ID (the unique ID of a node); 2) Node type (upstream node or downstream node); 3) Destination address of each node (either upstream or downstream); 4) the port number of SD switch which is connected to the node (either upstream or downstream); 5) VLAN-tag used for the group (only necessary when the underlying network is Ethernet); and 6) TLP routing ID (the unique ID of TLP routing) illustrated in FIG.13 from the received packets.

The SDN box consists of at least one SDSW controller and at least one SD switch. The whole processing process is as follows. When an encapsulated packet

(whose inner packet is PCI-E packet) is received by the RcevD module 502 on a SD switch 501, it is decapsulated by the decapsulation module 505 to a PCI-E packet, the TLP routing ID is extracted from it. Then the slave TLP routing table operation module 508 will retrieve the slave TLP routing table based on the extracted TLP routing ID.

- If not found, the packet will be buffer at the packet buffer module 507 and then a query packet (contains the TLP routing ID, node type, node ID, as well as other related information illustrated as FIG.13) will be sent to the SDSW controller 601 by the query (SendC) module 504. At the side of the SDSW controller 601, the query is received by the reception (RecvSW) module 602. The master TLP routing table on the SDSW controller 601 will be further retrieved by the master TLP routing table operation module 604 based on the TLP routing ID.
  - If not found, the master TLP routing table operation module 604 on the SDSW controller 601 will create a new table item based on extracted info by the msg-parser module 605, which includes: 1) Node ID (the unique ID of a node); 2) Node type (upstream node or downstream node); 3) Destination address of each node (either upstream or downstream); 4) the port number of SD switch which is connected to the node (either upstream or downstream); 5) VLAN-tag used for the group (only necessary when the underlying network is Ethernet); and 6) TLP routing ID (the unique ID of TLP routing) illustrated in FIG.13. Then it is inserted into the master TLP routing table. The sending (SendSW) module 603 on the SDSW controller 601 will notify the SD switch 501 to broadcast the PCI-E-over-Ethernet packet.
  - If found, the sending (SendSW) module 603 on the SDSW controller 601 will notify the SD switch 501 to forward the encapsulated PCI-E packet

according to the retrieved destination address. The notification is processed at the RecvC module 509 and the same new table item is inserted into the slave TLP routing table by the slave TLP routing table operation module 508 on SD switch 501.

- If found, the PCI-E packet will be encapsulated by the encapsulation module 506 with an encapsulation packet header, which is filled with the retrieved address as the destination address returned by the slave TLP routing table operation module 508.

<DEVINFO packet format>

During the whole communication process between upstream node and downstream node, there are two kinds of packets are used. One kind of packet is PCI-E packet, which is used during the operation of memory read/write, I/O read/write, config read/write. The other kind packet is called DEVINFO packet, which is newly defined in our invention. It is used to: 1) start the communication between upstream node and downstream node; and 2) remind each other they are keeping alive periodically at a certain interval. The packet format of DEVINFO at least contains data fields (but not limited to it) as follows: 1) Node ID (the unique ID of a node); 2) Node type (upstream node or downstream node); 3) Source address of the node (either upstream or downstream); and 4) TLP routing ID (the unique ID used in TLP routing). FIG. 7 is a table depicting an embodiment of a possible definition of the packet format of the DEVINFO packet.

<Communication between an upstream node and a downstream node>

FIG. 8 is a sequence diagram of the memory read/write, (the process of I/O read/write, config read/write follows the same sequence diagram, so that it is omitted) between an upstream node 801, SDN box 802 and a downstream node 803. The communication process is as follows.

① When the upstream node 801 accesses memory (e.g. memory read/write operation) of the downstream node 803, it sends PCI-E packets. The PCI-E packets are encapsulated with a packet header in step 804 by the encapsulation/decapsulation module 303 and sent out from the NIC 304.

② The encapsulated packets arrive at a certain port of a SD switch in the SDN box 802. The encapsulated packets are decapsulated in step 805, and then their inner information is extracted in step 806. Then the decapsulated packets are encapsulated in step 807 and forwarded to the downstream node 803.

③ When the encapsulated packets are received at the downstream node 803, they are decapsulated to PCI-E packets in step 808.

The process of communication from the downstream node to the upstream node (e.g. replying the result of memory read) is the same as the above steps. The diagram is omitted.

FIG. 9 is a flowchart depicting an embodiment of a method of an upstream node or a downstream node how sends out PCI-E packets.

① When a node (either upstream node or downstream node) wants to send out packets in step 901, if it is a DEVINFO packet in step 902, the destination address (in the encapsulation packet header) should be a broadcast address in step 903.

② Otherwise it must be a PCI-E packet; the destination address (in the encapsulation packet header) should be in a predefined format, which can be recognized by the SD switch in step 904.

③ Finally, the packet is encapsulated with the outer packet header, where the destination address is the result from step 903 and step 904. The type of the outer packet header depends on the underlying network. For instance, if the underlying network is Ethernet, an Ethernet packet header is to be added as the outer packet

header and sent out in step 905.

FIG. 10 is a flowchart depicting an embodiment of a method of an upstream node or a downstream node how receives encapsulated packets.

① When receives an encapsulated packet in step 1001, the packet will be decapsulated into a PCI-E packet in step 1002.

FIG. 11 is a flowchart depicting an embodiment of a method of the SD switch how to handles encapsulated packets. The process of the SD switch is as follows.

① If there is not a packet from SDSW controller in step 1101, the packet is checked whether belongs to the specific encapsulated packet in step 1104. If it is not, the packet will be processed further in the routine of other packets in step 1105.

② If it is the encapsulated packets from upstream node or downstream node, its TLP routing ID is extracted and the slave TLP routing table will be retrieved based on it in step 1106. If not found, the packet will be buffered, a query request is sent to SDSW controller in step 1108. If found, the packet will be encapsulated with an encapsulation header, where the retrieved address is filled in as the destination address, and sent out in step 1107.

③ If there is a packet received from SDSW controller in step 1101, the carried TLP routing info is extracted in step 1102, and adding the new table item to the slave routing table in step 1103. Then the previous buffered packet will be further processed (encapsulates the packets, fills in the retrieved destination address and sends it out) in step 1107.

FIG. 12 is a flowchart depicting an embodiment of a method of the SDSW controller how to process the query request from the SD switch, extract TLP info and update the master TLP routing table, and finally notify SD switch to update the slave TLP routing table. The process is as follows.

① If there is a query request packet received from SD switch in step 1201, the TLP routing information will be extracted in step 1202.

② The master TLP routing table will be retrieved based on the TLP routing information in step 1203. If nothing is found, the new item will be added to the master TLP routing table in step 1204.

③ Finally, the result of retrieval is sent out to notify SD switch in step 1205.

FIG.13 is a possible data structure of TLP routing table, which is used on both SD switch and SDSW controller, when the underlying network is Ethernet. The table consists of columns as follows: 1) Node ID (the unique ID of a node); 2) Node type (upstream node or downstream node); 3) Destination address of each node (either upstream or downstream); 4) the port number of SD switch which is connected to the node (either upstream or downstream); 5) VLAN-tag used for the group (only necessary when the underlying network is Ethernet); and 6) TLP routing ID (the unique ID of TLP routing). FIG.13 indicates that two groups of nodes, whose VLAN IDs are 1 or 2. The first three items belong to a group, because they share the same VLAN ID 1. The node whose node ID is 1, is an upstream node, whose MAC address is MAC_00. It is connected to the 1st port of the SD switch, and its TLP ID is bus0/dev0/func0. The node whose node ID is 2, is a downstream node, whose MAC address is MAC_01. It is connected to the 2nd port of the SD switch, and its TLP ID is bus1/dev1/func1.

<Failure detection and hand-over>

FIG. 14 is a sequence diagram of the failure detection and hand-over between an upstream node 1401, SD switch (e.g. OpenFlow switch but not limited to OpenFlow) 1402, SDSW controller (e.g. OpenFlow controller but not limited to OpenFlow) 1403 and a downstream node (it is omitted for clear explanation). The process of failure detection and hand-over is as follows.

① Once there is failure in step 1404 on the upstream node 1401, the link down network signal will be sent to the SD switch 1402.

② When the SD switch 1402 receives the link down network signal, it will notify the SDSW controller 1403 in step 1405. For an instance, in OpenFlow , the OpenFlow switch will send OFPPS_LINK_DOWN message to the OpenFlow controller .

③ When the SDSW controller 1403 receives the notification from the SD switch 1402, it will find another available upstream node to hand over in step 1406, modify the master TLP routing table of the connected downstream nodes in step 1407, and then notify the SD switch in step 1408.

④ When the SD switch 1402 receives the notification from the SDSW controller 1403, it will modify the slave TLP routing table in step 1409. So that the connected downstream nodes are handed-over to an upstream node with a new group ID

While preferred embodiments of the invention have been described and illustrated above, it should be understood that these are exemplary of the invention and are not to be considered as limiting. Additions, omissions, substitutions, and other modifications can be made without departing from the scope of the present invention. Accordingly, the invention is not to be considered as being limited by the foregoing description, and is only limited by the scope of the appended claims.

CLAIMS

1. A method of using a software-defined network (SDN) box as a peripheral component interconnect-express (PCI-E) switch over a network which including at least one upstream node, at least one downstream node and a SDN box, the method comprising:

interconnecting the upstream node and the downstream node via the network by the SDN box;

sending a PCI-E packet at one side of the upstream and downstream nodes;

encapsulating the PCI-E packet with a specific packet header;

sending to the network the encapsulated packets based on transaction layer packet (TLP) routing identification (ID) carried with inner PCI-E packet by the SDN box;

receiving the encapsulated packet at the other side of the upstream and downstream nodes;

removing the specific packet header from a received packet; and

decapsulating the received packet into the PCI-E packet.

2. The method according to claim 1, wherein:

the upstream node includes a computing unit (CPU), a network interface card (NIC) and an encapsulation/decapsulation module; and

the downstream node includes an I/O device, a network interface card (NIC) and an encapsulation/decapsulation module.

3. The method according to claim 1, wherein the upstream node and the

downstream node belong to at least one group including a single upstream node and at least one downstream node.

4. The method according to claim 1, wherein the TLP routing ID is the identification of TLP routing methods in the PCI-E including address routing, ID routing and implicit routing.

5. The method according to claim 1, wherein the SDN box is implemented as the format of a software-defined (SD) switch and a software-defined switch (SDSW) controller corresponding to the SD switch, the SD switch including a slave TLP routing table and the SDSW controller including a master TLP routing table.

6. The method according to claim 5, comprising:

with the SD switch, decapsulating the received packets from the one side of the upstream node and downstream node, extracting the TLP routing ID, retrieving the slave TLP routing table based on the TLP routing ID,

if a destination address is found in the TLP routing table, encapsulating the packet with a packet header, filling the destination address and sending out,

if a destination address is not found in the TLP routing table, buffering the packet, and a query is sending out to the SDSW controller; and

with the SDSW controller, parsing the query request from the SD switch, extracting TLP routing information, adding the TLP routing information to the master TLP routing table as a new table item and notifying the SD switch to update the slave TLP routing table.

7. The method according to claim 5, wherein the SD switch and the SDSW controller corresponding to the SD switch are connected with a communication channel, by which query and notify messages is transferred, the communication channel being either remote communication channel including Ethernet or local communication channel including UNIX domain socket.

8. The method defined in claim 1, wherein, in the step of encapsulating of PCI-E packets, the destination address is kept in a pre-defined format which can be recognized by the SDN box.

9. The method according to claim 5, when failure occurs, built-in notification of the SDN box triggers modification of the TLP routing table at the SD switch and the SDSW controller, which achieves handing over of the downstream node.

FIG. 1

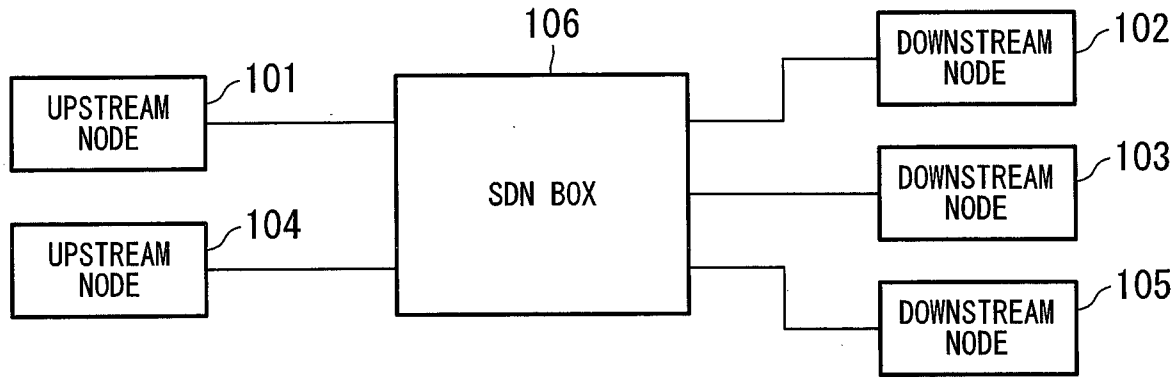

106
UPSTREAM NODE
101
UPSTREAM NODE
104
SDN BOX
DOWNSTREAM NODE
102
DOWNSTREAM NODE
103
DOWNSTREAM NODE
105

FIG. 2A

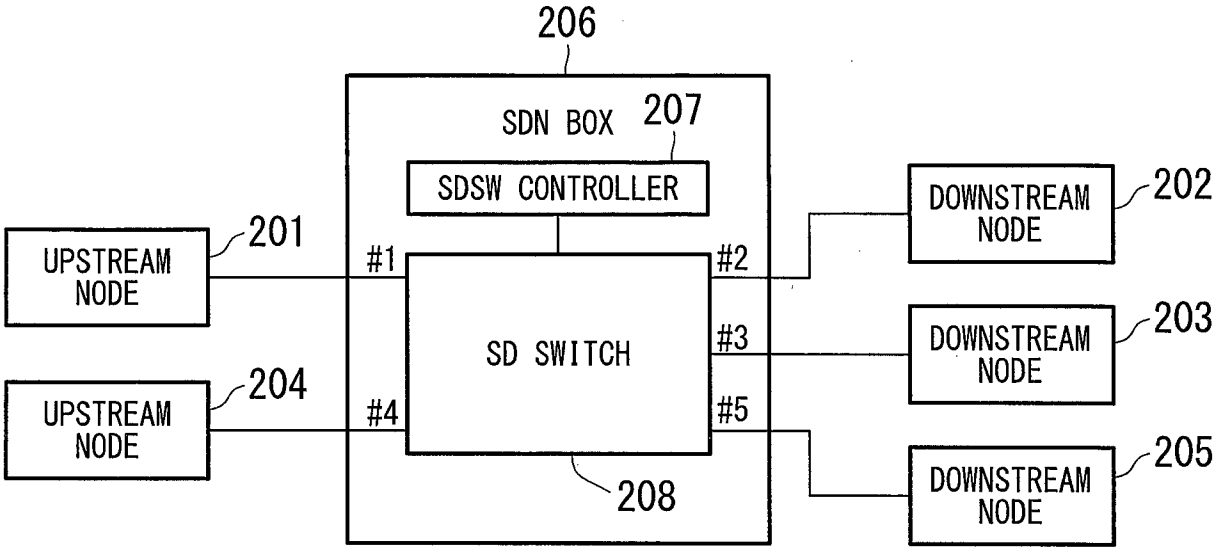

206
SDN BOX
207
SDSW CONTROLLER
UPSTREAM NODE
201
UPSTREAM NODE
204
#1
#4
SD SWITCH
#2
#3
#5
208
DOWNSTREAM NODE
202
DOWNSTREAM NODE
203
DOWNSTREAM NODE
205

FIG. 2B

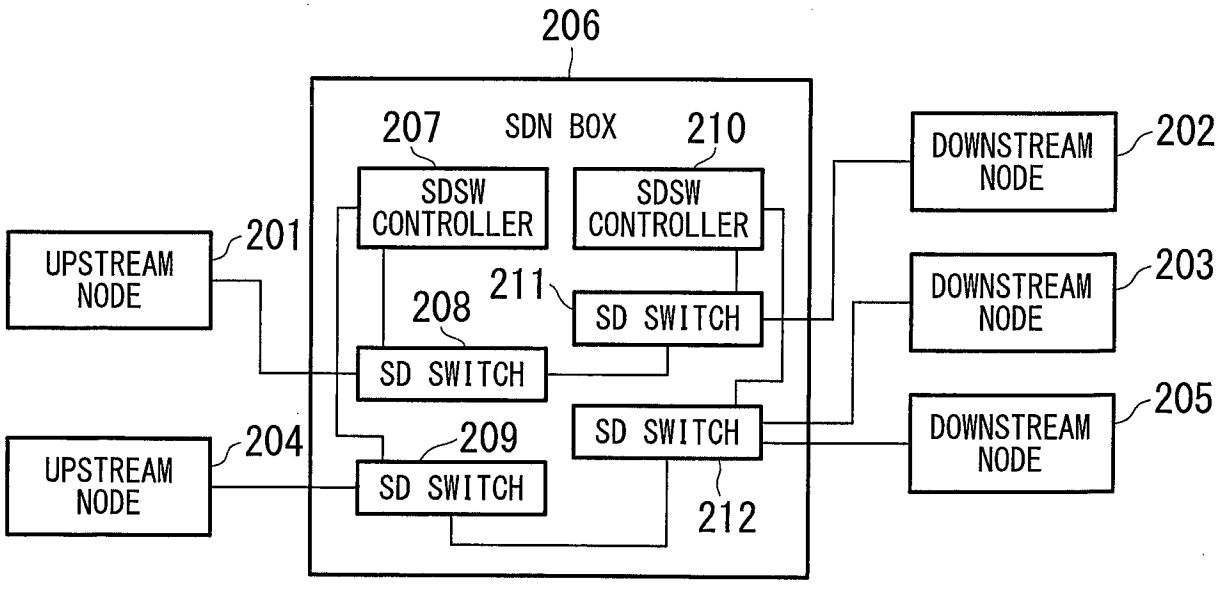

206
207
SDN BOX
210
SDSW CONTROLLER
SDSW CONTROLLER
UPSTREAM NODE
201
UPSTREAM NODE
204
208
211
SD SWITCH
SD SWITCH
209
SD SWITCH
SD SWITCH
212
DOWNSTREAM NODE
202
DOWNSTREAM NODE
203
DOWNSTREAM NODE
205

FIG. 3

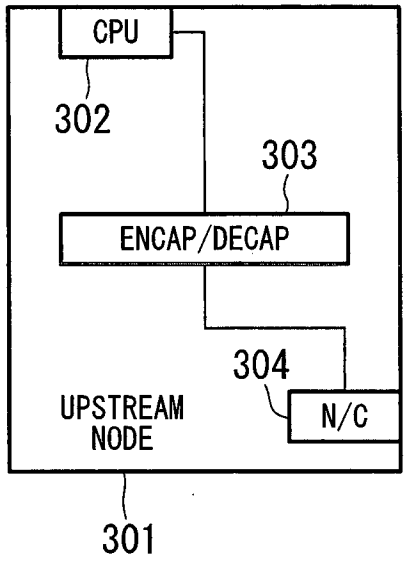

CPU
302
303
ENCAP/DECAP
304
N/C
UPSTREAM NODE
301

FIG. 4

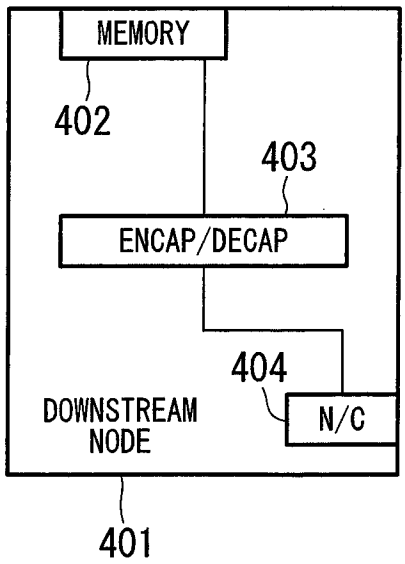

MEMORY
402
403
ENCAP/DECAP
404
N/C
DOWNSTREAM NODE
401

FIG. 5

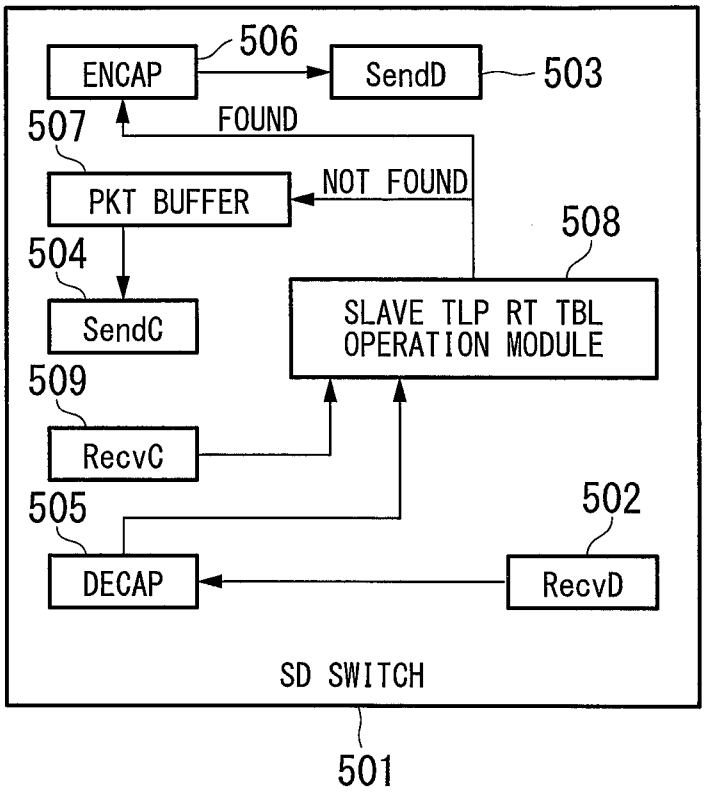

ENCAP
506
SendD
503
FOUND
507
PKT BUFFER
NOT FOUND
508
504
SendC
SLAVE TLP RT TBL
OPERATION MODULE
509
RecvC
505
502
DECAP
RecvD
SD SWITCH
501

FIG. 6

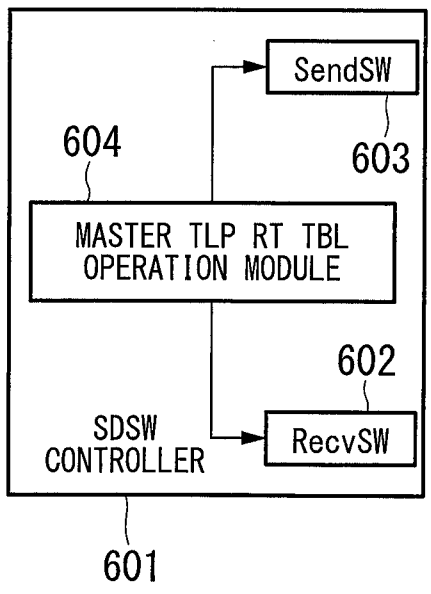

SendSW
604
603
MASTER TLP RT TBL
OPERATION MODULE
602
SDSW
CONTROLLER
RecvSW
601

FIG. 7

NODE ID	TYPE	SRC ADDRESS	TLD ROUTING ID
44	4	48	32

(bit)

FIG. 8

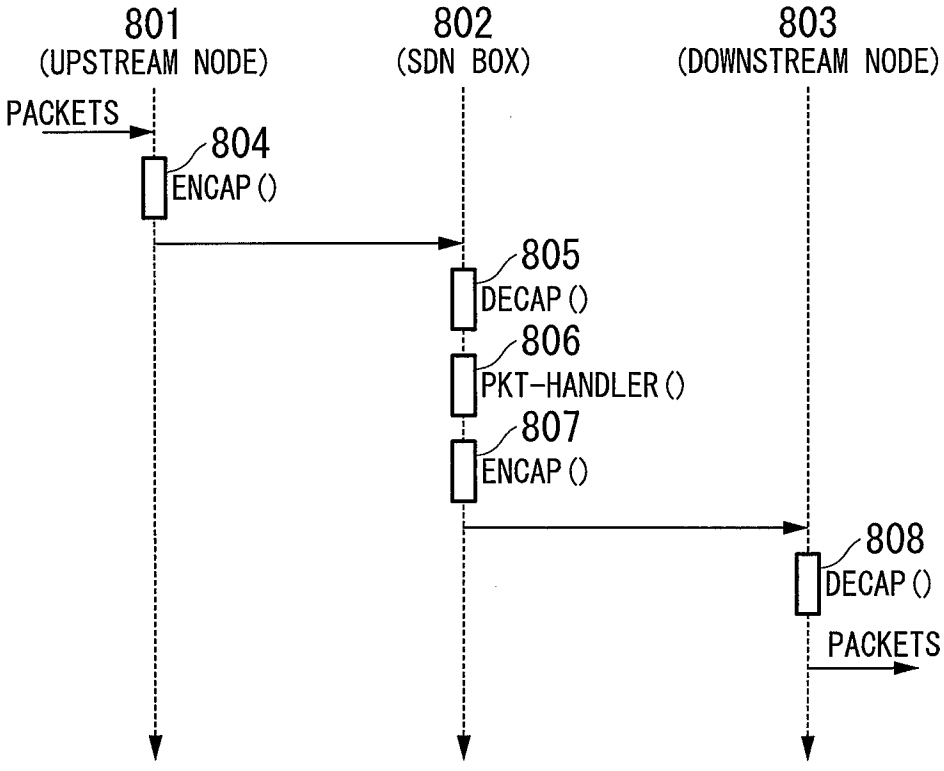


FIG. 9

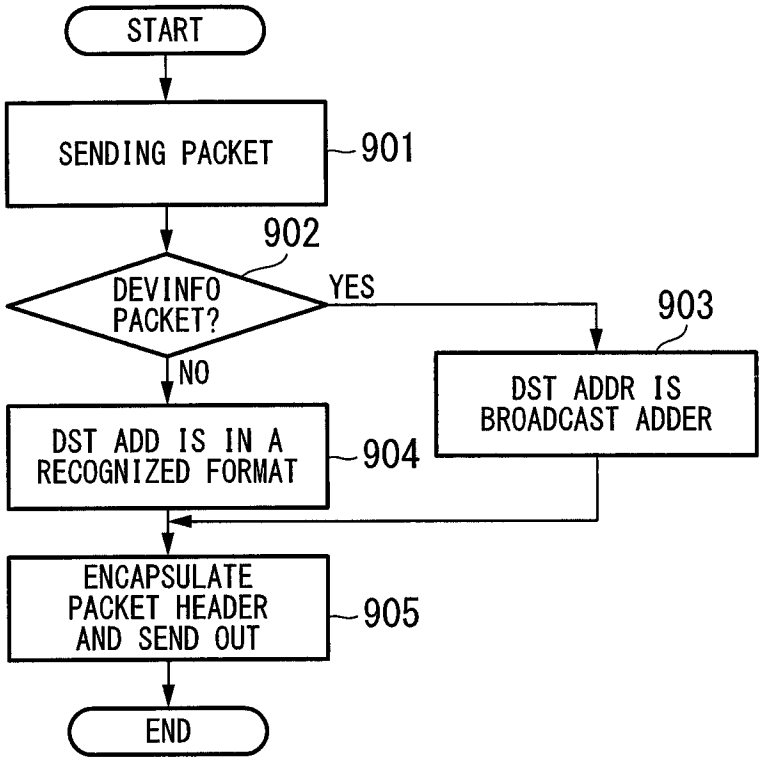

START
SENDING PACKET
901
902
DEVINFO PACKET?
YES
903
DST ADDR IS BROADCAST ADDER
NO
DST ADD IS IN A RECOGNIZED FORMAT
904
ENCAPSULATE PACKET HEADER AND SEND OUT
905
END

FIG. 10

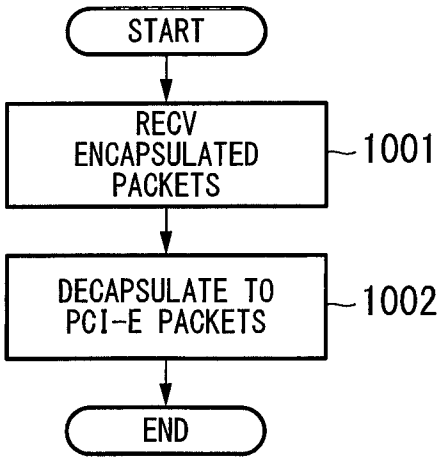

START
RECV ENCAPSULATED PACKETS
1001
DECAPSULATE TO PCI-E PACKETS
1002
END

FIG. 11

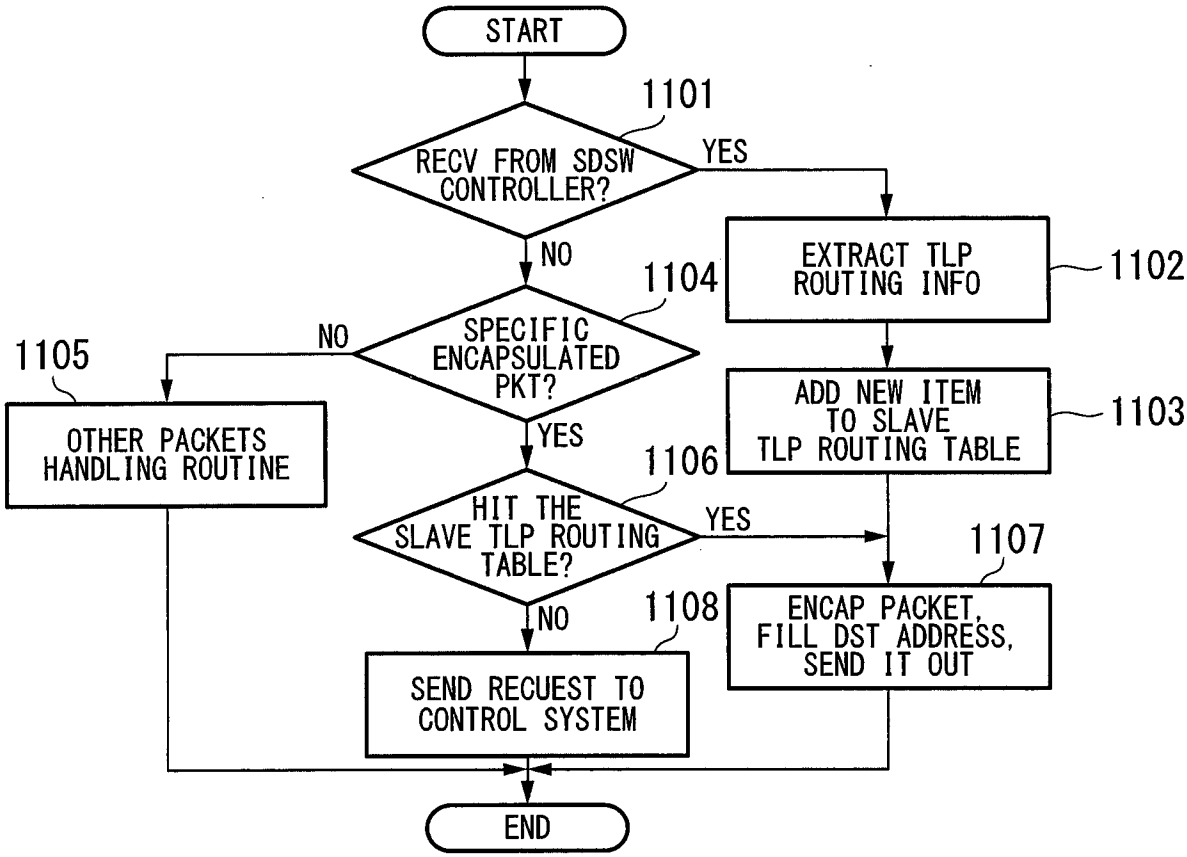

START
1101
RECV FROM SDSW CONTROLLER?
YES
NO
EXTRACT TLP ROUTING INFO
1102
1104
SPECIFIC ENCAPSULATED PKT?
NO
1105
OTHER PACKETS HANDLING ROUTINE
YES
ADD NEW ITEM TO SLAVE TLP ROUTING TABLE
1103
1106
HIT THE SLAVE TLP ROUTING TABLE?
YES
1107
NO
1108
ENCAP PACKET, FILL DST ADDRESS, SEND IT OUT
SEND RECUEST TO CONTROL SYSTEM
END

FIG. 12

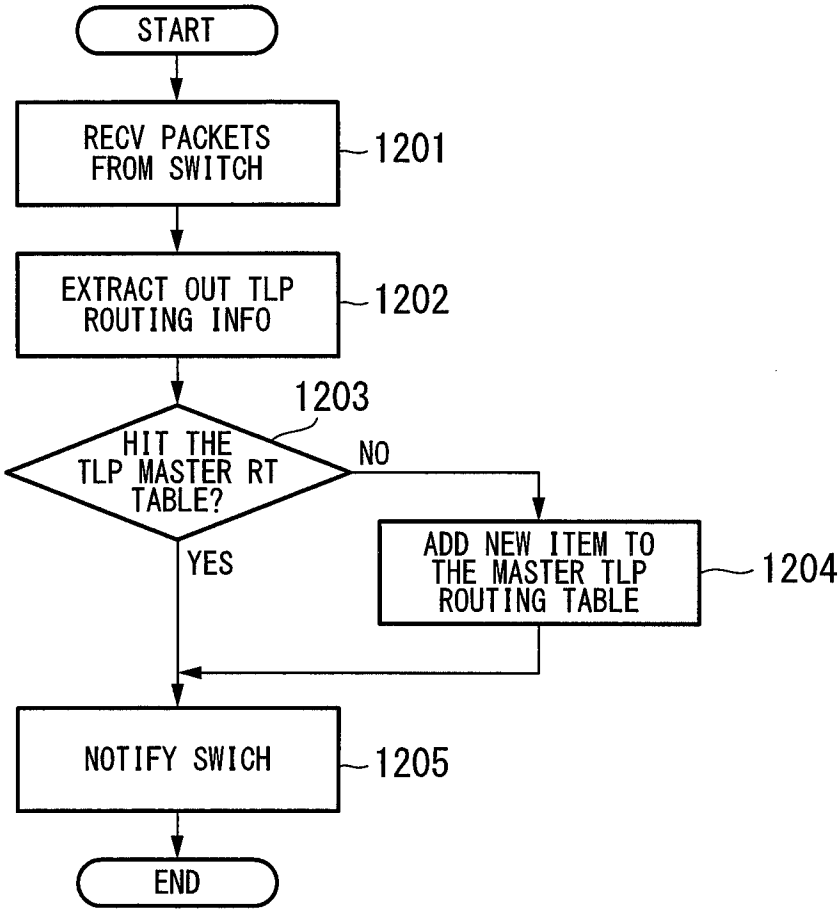
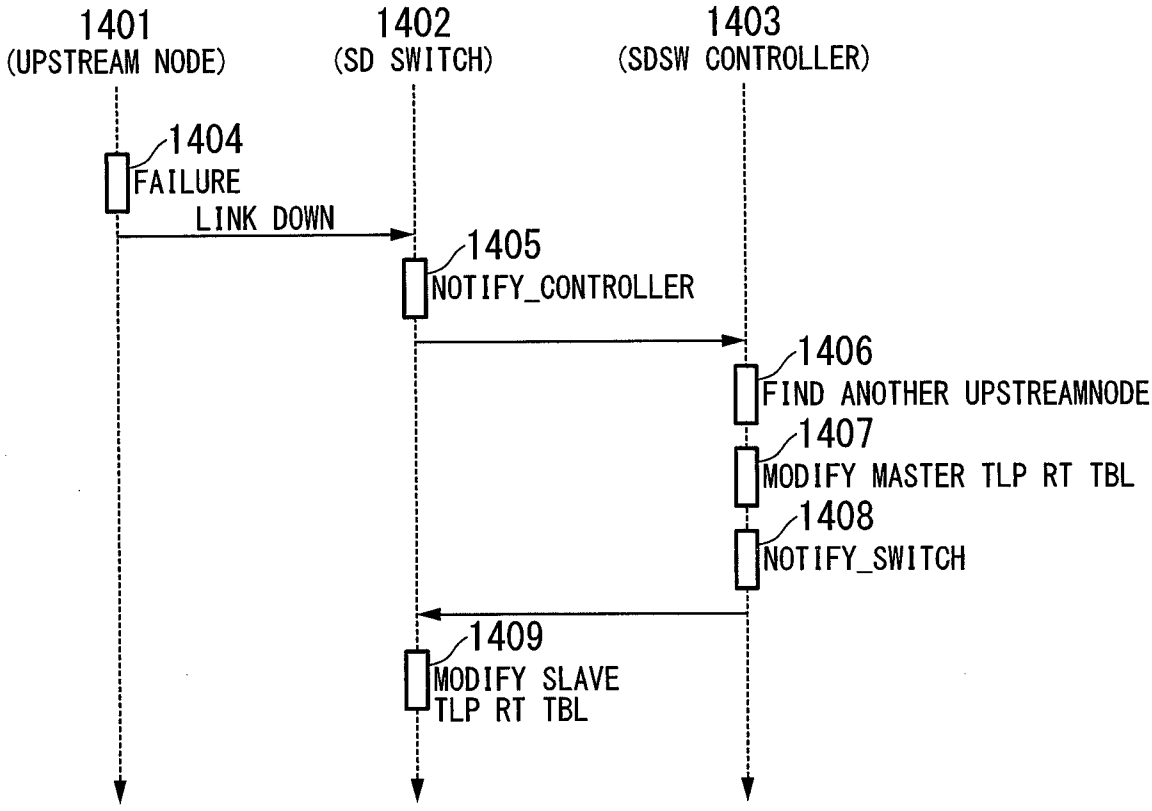


FIG. 13

NODE ID	NODE TYPE	DEST_ADDER	PORT NO.	VLAN ID	TLP ID
1	UPSTREAM	MAC_00	1	1	BUS0/DEV0/FUNC0
2	DOWNSTREAM	MAC_01	2	1	BUS1/DEV1/FUNC1
3	DOWNSTREAM	MAC_02	3	1	BUS2/DEV2/FUNC2
4	UPSTREAM	MAC_10	4	2	BUS0/DEV0/FUNC0
5	DOWNSTREAM	MAC_11	5	2	BUS1/DEV1/FUNC1
6	DOWNSTREAM	MAC_12	6	2	BUS2/DEV2/FUNC2

FIG. 14


1401
(UPSTREAM NODE)
1402
(SD SWITCH)
1403
(SDSW CONTROLLER)
1404
FAILURE
LINK DOWN
1405
NOTIFY_CONTROLLER
1406
FIND ANOTHER UPSTREAMNODE
1407
MODIFY MASTER TLP RT TBL
1408
NOTIFY_SWITCH
1409
MODIFY SLAVE
TLP RT TBL

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2014/058146

A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl. G06F13/36(2006.01)i, G06F13/14(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl. G06F13/10, G06F13/38, H04L12/00, H04W

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Published examined utility model applications of Japan 1922-1996
Published unexamined utility model applications of Japan 1971-2014
Registered utility model specifications of Japan 1996-2014
Published registered utility model applications of Japan 1994-2014

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	JP 2007-219873 A (NEC Corporation) 2007.08.30, paragraphs [0020] to [0082], figure 1 to 17 & US 2007/0198763 A1 & US 2011/0153906 A1	1-9
Y	WO 2013/051386 A1 (NEC Corporation) 2013.04.11, whole document, figure 1 to 7 (Family: none)	1-9
Y	JP 2014-003392 A (NTT DOCOMO, INC) 2014.01.09, paragraphs [0051] to [0052], figure 1 to 2 (Family: none)	9

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
02.06.2014	10.06.2014

Name and mailing address of the ISA/JP	Authorized officer
Japan Patent Office 3-4-3, Kasumigaseki, Chiyoda-ku, Tokyo 100-8915, Japan	Masaaki Kokawa 5T 3242 Telephone No. +81-3-3581-1101 Ext. 3568