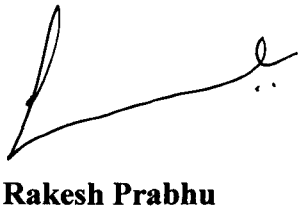
ABSTRACT

The embodiments of the present invention provide a system and method for providing multi-processor data plane architecture. The method for comprises selecting several processing units for performing an egress data process and an ingress data process. A number of processing units is selected for performing an egress data process based on a type of switch and data rate. Several processing pipes are provided for each processing unit. Each processing pipe is divided into several processing stages based on a number of lookup tables used in the egress data process and the ingress data process to absorb a response time of a memory device. The data is stored in several databases in each processing unit, and the databases are copied into several banks to increase an access time with a storage device, like DDR-SDRAM. Several headers are resynchronized using a fixed delay time through an ACL unit.



Date: 28-03-2014
Place: Bangalore

Rakesh Prabhu
Counsel for Applicant

CLAIMS

What is claimed is:

1. A method for generating and managing dynamic data plane for MEF switches, MPLS-TP router in FPGA platform, the method comprises steps of:

selecting a plurality of processing units for performing an ingress data process and an egress data process, and wherein a number of processing units is selected for performing an ingress data process and an egress data process based on a type of switch and data rate;

providing a plurality of processing pipes for each processing unit;

dividing each processing pipe into a plurality of processing stages to absorb a response time of a memory device, and wherein each processing pipe is divided into the plurality of processing stages based on a number of lookup tables used in the ingress data process and the egress data process;

storing a plurality of data in a plurality of databases provided in each processing unit, and wherein the plurality of databases are copied into a plurality of banks to increase an access time with a storage device, and wherein the storage device is DDR-SDRAM; and

resynchronizing a plurality of headers using a fixed delay time through a lookup unit.

2. The method according to claim 1, wherein an ingress data processing comprises the steps of:

assigning a plurality of ports to each processing unit by an ingress arbiter switch matrix;

splitting a stored data across a plurality of processing pipes using a WRR block to reduce a processing load of each processing unit;

attaching a sequence number and a time stamp to each data frame;

copying a data into a temporary buffer, and wherein the temporary buffer is an ingress data buffer, and wherein the ingress data buffer is a slotted memory with buffers having a size of 128/256 bytes;

copying a header of preset size to each processing pipe as header queue;

performing a processing of the headers after a completion of the full header in the plurality of processing stages, and wherein a final processing stage is started after an elapse of an fixed delay to ensure that a processing of data and frames in all the processing stages are completed in a preset interval of time; and

updating a statistics block for a frame information and updating a policer for remaining credit.

3. The method according to claim 1, wherein an ingress data processing comprises:

extracting all header data in a first stage of processing, and wherein a frame processing is completed in a preset interval of time, and wherein a plurality of look-up tables is generated in the first processing stage, and wherein the plurality of look-up table includes a first table for port

database for admission and handling, a FPCR look-up table for a virtual database, and an ACL table;

collecting FPCR details from DDR in the second processing stage to perform FDB, tunnel lookups and learning validation; and

creating a meta data header based on all look-up tables and forwarding the created metadata to IDB for forwarding a frame along with a header to a traffic manager.

4. The method according to claim 1, wherein an egress data processing comprises the steps of:

forwarding a frame received from the traffic manger to individual egress data buffers (EDB), and wherein EDB is arranged for each processing unit in egress process;

copying meta data separately to a control buffer;

generating a memory request for a frame processing;

processing the frame in a freely available processing unit after receiving a response for the generated memory request;

forwarding the processed frame to individual ports after all checks; and

performing a resynchronizing operation of a data, and wherein the resynchronization of the data is done in a plurality of ports, when a speed of port is more than 10G.

5. A system for generating and managing dynamic data plane for ingress processing in switches, MPLS-TP router in FPGA platform, the system comprising:

an ingress arbiter;

a plurality of databases;

a plurality of ingress processing units, and wherein each ingress processing unit comprises a header queue, an ingress data buffer, a logic unit, a plurality of processing pipes; and

a traffic manager interface.

6. The system according to claim 5, wherein the logic unit comprises an address learning and ageing module, database multi-copy logic module, memory controller, flow statistics module, BRAM based database tables, ACL block, memory arbiter, and policing and statistics module.
7. The system according to claim 5, wherein the ingress data processing unit extracts all header data in a first stage of processing, and wherein a frame processing is completed in a preset interval of time, and wherein a plurality of look-up tables is generated in the first processing stage, and wherein the plurality of look-up table includes a first table for port database for admission and handling, a FPCR look-up table for a virtual database, and an ACL table, and wherein the ingress data processing unit collects FPCR details from DDR in the second processing stage to perform FDB, tunnel lookups and learning validation, and wherein the ingress data processing unit creates a meta data header based on all look-up tables and forwarding the created metadata to IDB for forwarding a frame along with a header to a traffic manager.

8. A system for generating and managing dynamic data plane for egress processing in switches, MPLS-TP router in FPGA platform, the system comprising:

a database table;

a memory controller;

a statistics interface;

an ingress arbiter;

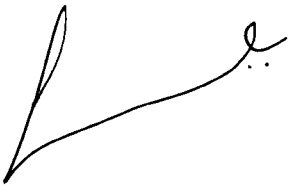
a plurality of egress processing units; and

a traffic manager interface.

9. The system according to claim 8, wherein the egress processing unit comprises a first buffer interface, a second buffer interface, data combining module, an egress checking module, a plurality of header modification modules, a memory request queue, a plurality of processing pipes, and egress data buffer.

10. The system according to claim 8, wherein the egress processing unit forwards a frame received from the traffic manger to individual egress data buffers (EDB), and wherein the egress processing unit copying meta data separately to a control buffer, and the egress processing unit generating a memory request for a frame processing, and wherein the egress processing unit processes the frame when the processing unit is freely available for a processing operation after receiving a response for the generated memory request, and wherein the egress processing unit forwards the processed frame to individual ports after all checks; and

where the egress processing unit performs a resynchronizing operation of a data, and wherein the resynchronization of the data is done in a plurality of ports, when a speed of port is more than 10G.



Rakesh Prabhu
Patent Agent,
FormulateIP Technolegal Solutions Private Limited,
#677, 1st Floor, 27th Main, 13th Cross, HSR Layout, Sector - 1
Bangalore – 560068, Karnataka, India

To,
The Controller of Patents,
The Patent Office, At Chennai

Applicant: TEJAS NETWORKS LIMITED
Application No:

No. of Sheets: 4
Sheet No: 1 of 4

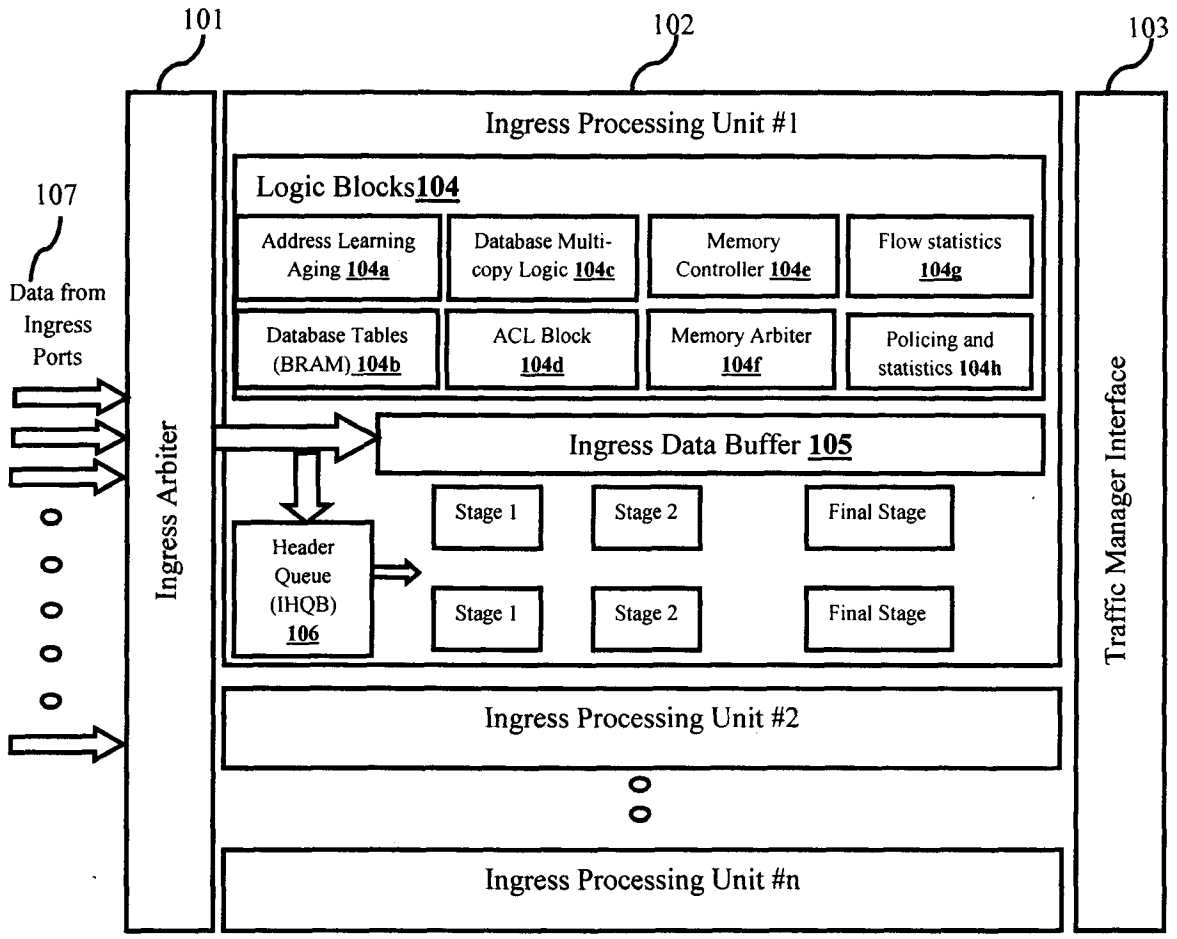


FIG. 1

Date: 28-03-2014
Place: Bangalore

Rakesh Prabhu
Counsel for Applicant

2/4

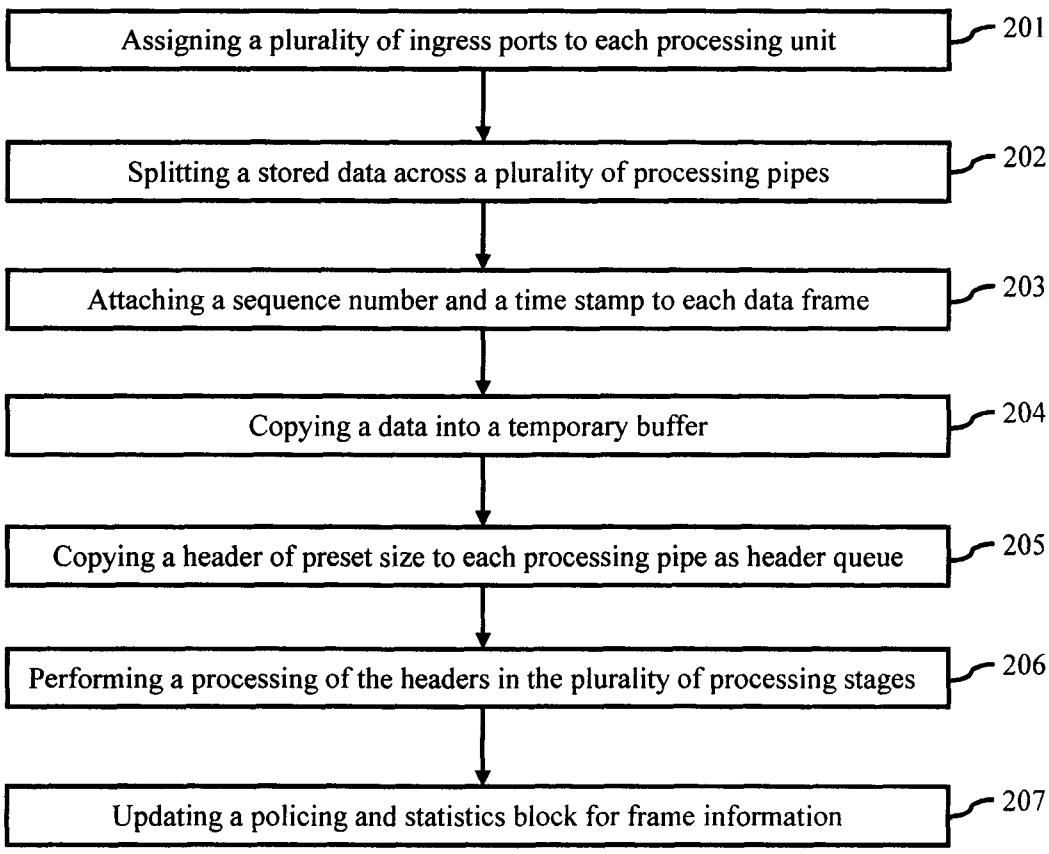


FIG. 2

Date: 28-03-2014
Place: Bangalore

Rakesh Prabhu
Counsel for Applicant

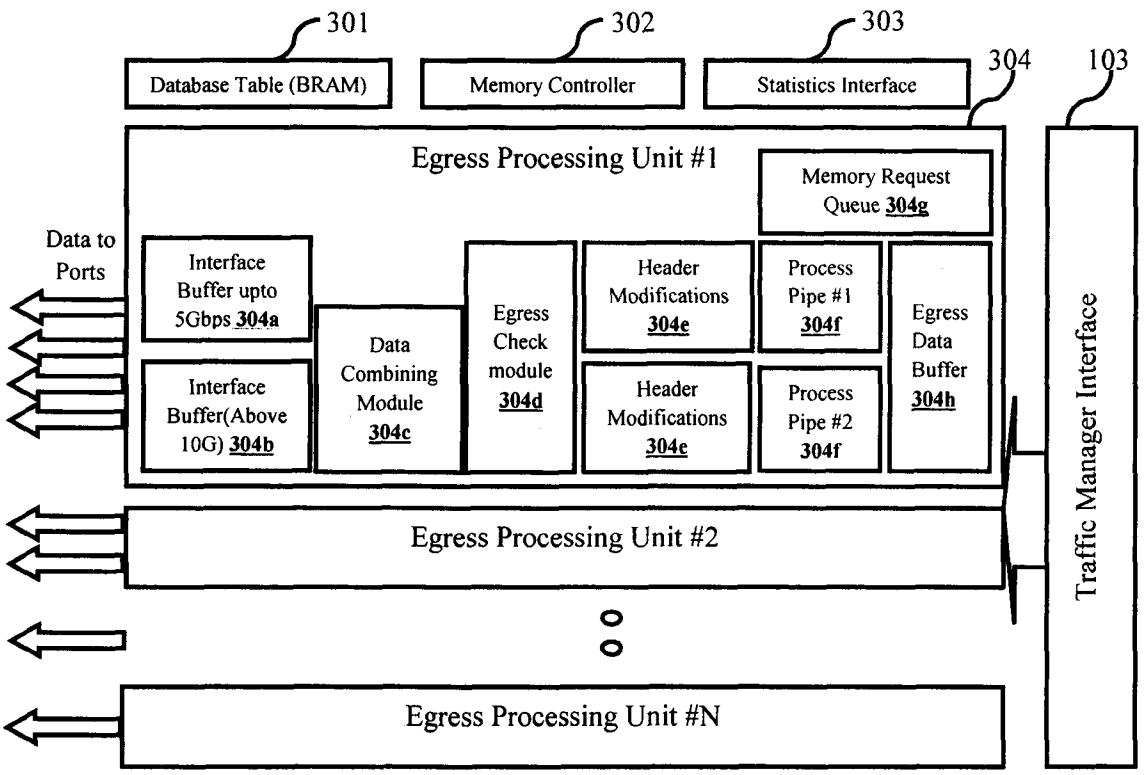


FIG. 3

Date: 28-03-2014

Place: Bangalore

Rakesh Prabhu

Counsel for Applicant

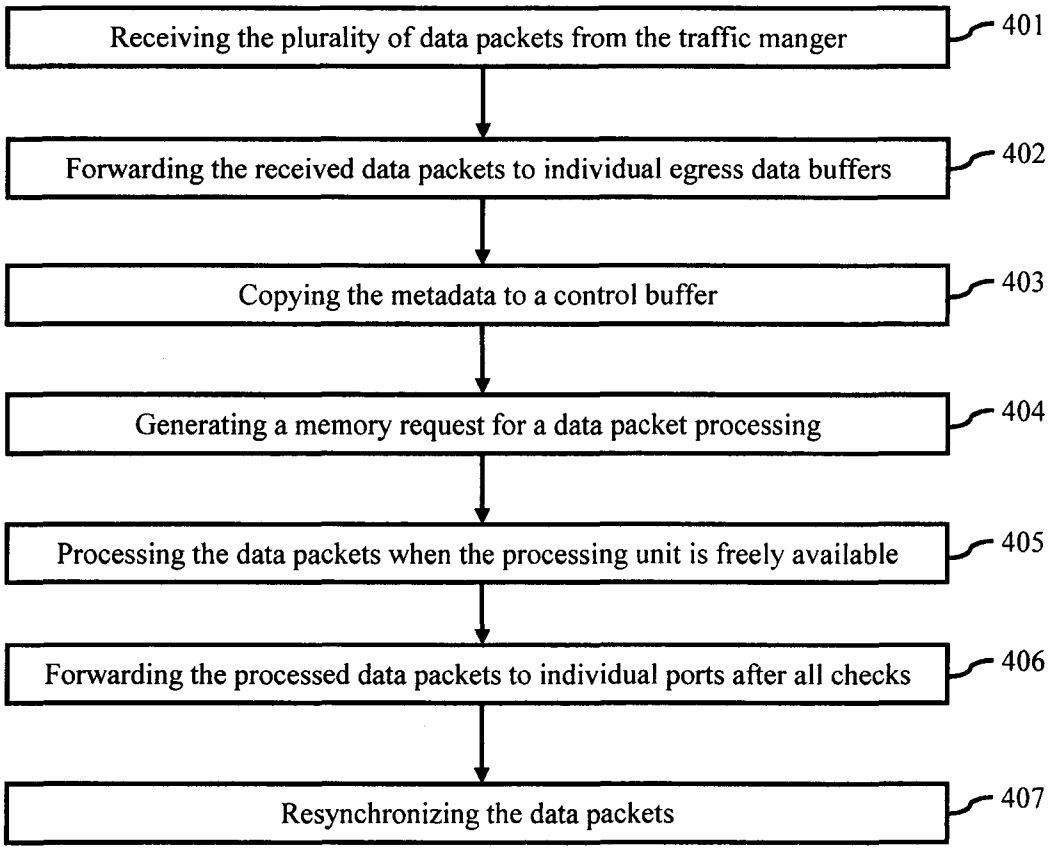


FIG. 4

Date: 28-03-2014
Place: Bangalore

Rakesh Prabhu
Counsel for Applicant

A) TECHNICAL FIELD

[0001] The present invention generally relates to data plane architecture. The present invention particularly relates to an Ethernet data plane architecture for switching and routing, which scales across multiple design requirements of a communication network. The present invention more particularly relates to a dynamic data plane architecture for implementation of Metro Ethernet Forum [MEF] switches, MPLS-TP and router design on FPGA platform.

B) BACKGROUND OF THE INVENTION

[0002] A data plane is a part of switch/router architecture which supports the virtual switching or routing needs. Based on the needs of a particular customer, the switch employed in the data plane must be able to support the speeds ranging from 5G to 100G speed [or more]. The switches must also be able to support the speeds of the various ports and configurations which are in the range of 2 Mbps [TDM emulation over Ethernet] to 10Gbps and at times more. The port interface must further support Ethernet at 10/100/1000 and 10G, multiport EoS interface and Modem interface. The hardware cost is low for an aggregate switch capacity of 5 Gbps, while an expensive and large FPGA is required for 100 Gbps switch. For example, the data bus width shall be 1024 bits at a clock rate of 156MHz or 512 bits at an operating frequency of 250 MHz to design a switch data plane which supports 100Gbps. It is very difficult to meet the timings in FPGA at 250MHz, especially when the utilization crosses 70%, while it is easy and simple to

proceed with the 1024 bit data bus. However, a routing of 1024 line logic needs an expensive FPGA. Also the bus is inefficient to handle different frame size bursts. So the cost per port goes multiple times the expected cost per port, if the customer plans to use the same logic for a 16 Gbps switch. At 16 Gbps, bus width shall be 128 bits at clock of 175MHz. Further, multiple FPGAs are employed for designing a fast switching data plane. In view of foregoing, there is a need for a data plane architecture that scales across the cost and size of the FPGA devices as well as the architecture has to be flexible to scale across multiple FPGAs. The architecture must be adapted to meet the various sizes and speeds without any additional cost at slower speed or compromising the performance at high speed.

[0003] The dataplane mainly decides on the routing of incoming packets depending on a lookup table. The router looks up the destination address of the incoming packet and retrieves the information necessary to determine the path from the receiving element, through the internal forwarding switch of the router, and to the proper outgoing interface(s). Metro switch design needs multiple and large lookup tables to store multiple flow points and MAC address database. For storing the lookup tables, the dataplane requires several database accesses of different sizes and interfaces to different port speeds. Normal DDR3 devices are of low in cost but the big delays in random access memory make them unusable. But using any other expensive memory [CAM, QDR SRAM etc] makes the product cost abysmally high. Hence there

is a need for a dataplane architecture that adopts different memory types to store the various databases.

[0004] The above mentioned shortcomings, disadvantages and problems are addressed herein and which will be understood by reading and studying the following specification.

C) OBJECT OF THE INVENTION

[0005] The primary object of the present invention is to provide a system and method for generating and managing dynamic dataplane for MEF switches, MPLS-TP router in FPGA platform,

[0006] Another object of the present invention is to provide a flexible dataplane architecture that scales across the various ranges of sizes and speeds of a FPGA.

[0007] Another object of the present invention is to provide a multi-processing pipe architecture that scales across the multiple FPGAs to provide a fast switching dataplane.

[0008] Yet another object of the present invention is to provide a dataplane architecture that adopts over different memory types in order to store the multiple lookup tables and databases.

[0009] These and other objects and advantages of the present invention will become readily apparent from the following detailed description taken in conjunction with the accompanying drawings.

D) SUMMARY OF THE INVENTION

[0010] The various embodiments of the present invention provide a system and method for providing a multi-processing dataplane architecture. The dataplane architecture is flexible across multiple FPGA architectures and the architecture seamlessly adapts to the size and speed without adding to the cost at slower speed or performance at high speed. The method for generating and managing a dynamic data plane for MEF switches, MPLS-TP router in FPGA platform comprises the steps of selecting a plurality of processing units for performing an egress data process and an ingress data process. A number of processing units is selected for performing an egress data process based on a type of switch and data rate. A plurality of processing pipes is provided for each processing unit. Each processing pipe is divided into a plurality of processing stages to absorb a response time of a memory device. Each processing pipe is divided into the plurality of processing stages based on a number of lookup tables used in the egress data process and the ingress data process. A plurality of data is stored in a plurality of databases provided in each processing unit, and the plurality of databases is copied into a plurality of banks to increase an access time with a storage device, which is DDR-SDRAM. A plurality of headers is resynchronized using a fixed delay time through an ACL unit.

[0011] The various embodiments of the present invention provide a method for an ingress data processing. The method comprises the steps of assigning a plurality of ports to each processing unit by an ingress arbiter switch matrix. A stored data is split across a plurality of processing pipes using

a weighted round robin [WRR] block to reduce a processing load of each processing unit. A sequence number and a time stamp are attached each data frame. A data is copied into a temporary buffer and the temporary buffer is an ingress data buffer. The ingress data buffer is a slotted memory with buffers having a size of 128/256 bytes. A header of preset size is copied to each processing pipe as header queue. The headers are processed in the plurality of processing stages after a completion of the full header and a final processing stage is started after an elapse of an fixed delay to ensure that a processing of data and frames in all the processing stages are completed in a preset interval of time. A statistics block is updated for a frame information and a policer is updated for remaining credit.

[0012] According to an embodiment of the present invention, the method for the ingress data processing further comprises following steps of extracting all header data in a first stage of processing. A frame processing is completed in a preset interval of time. A plurality of look-up tables is generated in the first processing stage. The plurality of look-up table includes a first table for admission and handling of port database, a FPCR look-up table for a virtual database, and an ACL table. The FPCR details are collected from DDR in the second processing stage to perform FDB, tunnel lookups and learning validation. A meta data header is created based on all look-up tables and the created metadata is forwarded to IDB for forwarding a frame along with a header to a traffic manager.

[0013] The various embodiments of the present invention provide a method for an egress data processing. The method comprises the following steps of forwarding a frame received from the traffic manger to individual egress data buffers (EDB), and wherein EDB is arranged for each processing unit in egress process. A meta-data header is copied separately to a control buffer. A memory request is generated for a frame processing. The frame is processed in a freely available processing unit after receiving a response for the generated memory request. The processed frame is forwarded to the individual ports after performing all the checks. The data is resynchronized and wherein the data is resynchronized in a plurality of ports, when a speed of port is more than 10G.

[0014] The various embodiments of the present invention provide a system for generating and managing a dynamic data plane for ingress processing in switches, MPLS-TP router in FPGA platform. The system comprises an ingress arbiter, a plurality of databases, a plurality of ingress processing units and a traffic manager interface. The ingress processing unit further comprises a header queue, an ingress data buffer, a logic unit, a plurality of processing pipes.

[0015] According to an embodiment of the present invention, the logic unit of the ingress processing unit comprises an address learning and ageing module, database multi-copy logic module, memory controller, a flow statistics module, BRAM based database tables, ACL block, memory arbiter, and policing and statistics module.

[0016] According to an embodiment of the present invention, the ingress data processing unit extracts all header data in a first stage of processing, and wherein a frame processing is completed in a preset interval of time. A plurality of look-up tables is generated in the first processing stage. The plurality of look-up table includes a first table for port database for performing admission and handling, a FPCR look-up table for a virtual database, and an ACL table. The ingress data processing unit collects FPCR details from DDR in the second processing stage to perform FDB, tunnel lookups and learning validation. The ingress data processing unit creates a meta data header based on all look-up tables and forwards the created metadata to IDB for forwarding a frame along with a header to a traffic manager.

[0017] The various embodiments of the present invention provide a system for generating and managing dynamic data plane for egress processing in switches, MPLS-TP router in FPGA platform. The system comprises a database table, a memory controller, a statistics interface, a plurality of egress processing units and a traffic manager interface.

[0018] According to an embodiment of the present invention, the egress processing unit comprises a first buffer interface, a second buffer interface, data combining/merging module, an egress checking module, a plurality of header modification modules, a memory request queue, a plurality of processing pipes, and egress data buffer.

[0019] According to an embodiment of the present invention, the egress processing unit forwards a frame received from the traffic manger to the

individual egress data buffers (EDB). The egress processing unit copies the meta data separately to a control buffer. The egress processing unit generates ing a memory request for a frame processing. The egress processing unit processes the frame when the processing unit is freely available for a processing operation after receiving a response for the generated memory request. The egress processing unit forwards the processed frame to the individual ports after a completion of all the checks. The egress processing unit resynchronizes a data in a plurality of ports, when a speed of port is more than 10G.

[0020] These and other objects and advantages of the present invention will become readily apparent from the following detailed description taken in conjunction with the accompanying drawings.

[0021] These and other aspects of the embodiments herein 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 descriptions, while indicating the preferred embodiments and numerous specific details thereof, are given by way of illustration and not of limitation. Many changes and modifications may be made within the scope of the embodiments herein without departing from the spirit thereof, and the embodiments herein include all such modifications.

E) BRIEF DESCRIPTION OF THE DRAWINGS

[0022] The other objects, features and advantages will occur to those skilled in the art from the following description of the preferred embodiment and the accompanying drawings in which:

[0023] FIG. 1illustratesa block diagram of a system for ingress data processing of incoming data packets, according to an embodiment of the present invention.

[0024] FIG. 2is a flowchart illustrating the steps involved in a method for ingress data processing of incoming data packets, according to an embodiment of the present invention.

[0025] FIG. 3illustrates a block diagram of a system illustrating egress data processing of frames received from the ingress data processing system through a traffic manager, buffer manager block, according to an embodiment of the present invention.

[0026] FIG. 4 is a flowchart illustrating the steps involved in a method for egress data processing of the data packets received from the ingress data processing system, according to an embodiment of the present invention.

[0027] Although the specific features of the present invention are shown in some drawings and not in others. This is done for convenience only as each feature may be combined with any or all of the other features in accordance with the present invention.

F) DETAILED DESCRIPTION OF THE INVENTION

[0028] In the following detailed description, reference is made to the accompanying drawings that form a part hereof, and in which the specific

embodiments that may be practiced is shown by way of illustration. These embodiments are described in sufficient detail to enable those skilled in the art to practice the embodiments and it is to be understood that the logical, mechanical and other changes may be made without departing from the scope of the embodiments. The following detailed description is therefore not to be taken in a limiting sense.

[0029] The various embodiments of the present invention provide a multi-processor architecture for a dataplane. The dataplane architecture is flexible across multiple FPGA architectures and the architecture seamlessly adapts to the sizes and speeds of multiple ports without adding to the cost for ports of lower speed or performance for ports of at high speed. The dataplane architecture comprises a plurality of processing units for performing an egress data process and an ingress data process. The plurality of processing units is configured for performing the egress data process and the ingress data process based on a type of switch and data rate of incoming data steam. Each data processing unit comprises a plurality of processing pipes and each processing pipe is further divided into a plurality of processing stages. The processing unit is divided into multiple processing stages based on a number of lookup tables used in the egress data process and the ingress data process. The processing unit further comprises a memory device. The memory device of the processing unit is configured for storing a plurality of databases and lookup tables. The memory device is any one of a Double Data Rate (DDR) - Synchronous Dynamic Random Access Memory (SDRAM), for example, DDR2 SDRAM,

DDR3 SDRAM and the like. The lookup tables comprise information corresponding to various routing tables, Ethernet flow point addresses and MAC addresses. The storage area of the memory device is divided into several banks thereby allowing the chip to work on several memory access commands at a time. The databases and lookup tables are copied into multiple banks, in-order to increase the access time with DDR-SDRAM. Other databases like Virtual Output Queue (VoQ) lookup, Virtual Local Area Network (VLAN) subscription, Port database, Policing, Protection, LAG, and Tunnel protocol and the like are stored in a block RAM. The processing stages facilitate the processing unit to absorb relatively high response time of the memory device.

[0030] According to an embodiment of the present invention, the dataplane switch is configured to use a single processing unit which is operating at a frequency of 125MHz for obtaining a speed of upto 8Gbps., The single processing unit is configured to operate at 200MHz frequency for obtaining a speed of upto 12G. Atleast two processing units are employed, for configuring a speed of upto 24Gbps. Atleast three processing units are employed to obtain a speed of 36G. Similarly atleast 8 processing units are employed for 100G switch.

[0031] According to an embodiment of the present invention, the datapath width at the processing unit level is typically 64 bits and the width is typically 32 bits at the processing pipe level in the ingress data processing stage. As VLAN, IP and TCP headers are aligned with 32 bit, the bus width is

limited to 32 bits. Further, the bus width is 32 bit at processing unit level in egress processing stage.

[0032] According to an embodiment of the present invention, the headers of each incoming data packet are resynchronized at the rate of a fixed delay through ACL section. The fixed delay is calculated as a sum of maximum delays of all database accesses and a margin time. The fixed delay is typically around 400 clocks at a frequency of 200MHz, whereas the delay is less for 125MHz operating frequency. The resynchronization of packet headers assists in re-ordering of the data packets according to the incoming order of data packets.

[0033] According to an embodiment of the present invention, the dataplane architecture comprises an ingress data processing system and an egress data processing system.

[0034] FIG. 1 illustrates a block diagram of a system for ingress data processing of incoming data packets, according to an embodiment of the present invention. The ingress data processing module comprises an ingress arbiter 101, a plurality of databases, a plurality of ingress processing units 102,a traffic manager interface103 and a plurality of ingress ports 107. The plurality of ingress processing units 102 further comprises an ingress header queue buffer 106, an ingress data buffer 105, a logic unit 104 and a plurality of processing pipes. The ingress arbiter 101 is configured to allocate a set of ingress ports 107 to each processing unit 102. When the speed of port is greater than a pre-determined threshold, the incoming data packets are directed to

multiple pipes. The data packets are split across multiple pipes according to a Weighted Round Robin (WRR) algorithm. At WRR block, a sequence number and a time stamp is attached with each data packet. The sequence number of each data packet identifies the incoming order of the respective data packet. The assignment of sequence number is irrespective of the port speed. The inclusion of the time stamp assists in IEEE1588 handling for synchronizing clocks throughout the dataplane. The splitting up of the data packets is carried out to maintain the port speed per processing unit at a level which is lower than a predetermined reference, for example, the speed must be atleast12.5Gbps at 200MHz or less than 6.5 Gbps at 125MHz. The port speed is typically a product of processing clock speed. The actual port clocks are not brought to processing level.

[0035] According to an embodiment of the present invention, the ingress data buffer 105 is configured to store the data from incoming data packets. The ingress data buffer 105 is typically a temporary buffer comprising a plurality of memory slots, where each memory slot is of fixed size. The incoming data packets are received at various ports of the ingress arbiter 101. Depending on the speed of incoming data packets, the ingress arbiter directs the incoming data packets to multiple ingress processing units 102. The received data packets at each ingress processing unit 102undergo processing at various stages. At initial stage, the data from the incoming data packets is stored in the ingress data buffer 105, whereas the header content of each data packet is extracted and copied to an Ingress Header Queue Buffer 106. The

processing of each data packet is completed within a fixed clock cycles. The ingress processing unit 102 further generates atleast three lookup tables. The various lookup tables comprise a port database lookup table, a Flow-Point Classification Rule (FPCR) lookup table, and an Access Control List (ACL) lookup table. The lookup table holding port database mainly supports information pertaining to admission of the ingress ports and handling of the data received on the ingress ports. The FPCR table provides a lookup for a virtual database. The ACL lookup table contains rules that are applied to port numbers or IP Addresses that are available on the ingress arbiter. Based on admission rules from the port database, a few non-matching data packets are marked as dropped packets, even though these packets are passed through the processing pipes to keep the distance same. The second processing stage collects FPCR details from the memory device to construct Forward DataBase (FDB), Tunnel lookup tables and learning validation. In the final processing stage, a metadata header is created based on the lookup tables of the plurality of databases. The created metadata header is forwarded to ingress data buffer 105, which in turn forwards the data packets along with the metadata headers to the traffic manager 106.

[0036] According to an embodiment of the present invention, the logic unit 104 of the ingress processing unit comprises an address learning and ageing module 104a, a database multi-copy logic module 104c, a memory controller 104e, a flow statistics module 104g, a BRAM based database tables 104b, an ACL block 104d, a memory arbiter 104f, and a policing and statistics

module 104h. The logic unit 104 is common across all the ingress processing units 102. As the data packets are forwarded to traffic manager interface 106, the flow statistics module 104g and the policing and statistics module 104h are updated with the information pertaining to data packets, where the information is related to both the ports and flow points/passing nodes. The policing and statistics module 104h is also updated with the remaining credits.

[0037] FIG. 2illustrates a flowchart explaining the steps involved in a method for the ingress data processing of incoming data packets, according to an embodiment of the present invention. The method comprises the following steps of assigning a plurality of ingress ports to each processing unit by an ingress arbiter switch matrix (201). A stored data is split across a plurality of processing pipes using a WRR block to reduce a processing load of each processing unit (202). A sequence number and a time stamp is attached to each data frame (203). A data is copied into a temporary buffer (204). The temporary buffer is an ingress data buffer. The ingress data buffer is a slotted memory with buffers having a fixed size of, for example, 128/256 bytes. A header of preset size is copied to each processing pipe as header queue (205). The headers are processed after a completion of the full header in the plurality of processing stages (206). A final processing stage is started after an elapse of an ACL delay to ensure that a processing of data and frames in all the processing stages are completed in a preset interval of time. A statistics block is updated with a frame information and a policy block is updated with the remaining credits (207).

[0038] FIG. 3 illustrates a block diagram of a system for egress data processing of the data packets received from the ingress data processing system, according to an embodiment of the present invention. The egress data processing system comprises a database table 301,a memory controller 302, a statistics interface 303, a plurality of egress processing units 304 and a traffic manager interface 103. The egress processing unit 304 further comprises a first buffer interface 304a, a second buffer interface 304b, data combining module 304c, an egress checking module 304d, a plurality of header modification modules 304e, a memory request queue 304g, a plurality of processing pipes 304f, and egress data buffer 304h. The data packets along with their respective metadata headers are received from the ingress data processing system through the traffic manager interface 103. The received data packets are forwarded to egress data buffers (EDB) 304h of individual egress processing unit. The egress processing unit 304 separately stores the metadata header into the control buffer 304h and a memory request is published at the memory request queue 304g for initiating a data packet processing operation. When a response for the memory request is received, the plurality of processing pipes 304f, which are freely available, process the data packets. The processed data packets are further forwarded to the individual output ports after undergoing various error verification processes. Since each processing pipe304f process a single data packet, the data packets needs to resynchronized before transmitting the data packets. The data packets are resynchronized based on the sequence number that is associated with each data packet to reorder the outgoing data

packets. The data packets are resynchronized when the speed of the port is more than a predefined threshold value, for example 100Gbps.

[0039] FIG. 4 illustrates a flowchart explaining a method of egress data processing of the data packets received from the ingress data processing system, according to an embodiment of the present invention. The method comprises following steps of receiving the plurality of data packets from the traffic manager (401). The received data packets are forwarded to the individual egress data buffers (EDB) (402). The metadata is copied to a control buffer by the egress processing unit (403). A memory request for a data packet processing is generated by the egress processing unit (404). The data packets are processed in the processing units that are freely available, after receiving a response for the generated memory request (405). The processed data packets is forwarding to individual ports after completing all the checks (406). The data packets are resynchronized according to the sequence number of the data packets, when the port speed is more than a predefined threshold value, for example 10G(407).

[0040] The foregoing description of the specific embodiments herein will so fully reveal the general nature of the embodiments herein that others can, by applying current knowledge, readily modify and/or adapt for various applications such specific embodiments herein without departing from the generic concept, and, therefore, such adaptations and modifications should and are intended to be comprehended within the meaning and range of equivalents of the disclosed embodiments. It is to be understood that the phraseology or

terminology employed herein is for the purpose of description and not of limitation. Therefore, while the embodiments herein have been described in terms of preferred embodiments, those skilled in the art will recognize that the embodiments herein can be practiced with modification within the spirit and scope of the appended claims.

G) ADVANTAGES OF THE PRESENT INVENTION

[0041] The various embodiments of the present invention provide a multi-processor architecture for a dataplane. The dataplane architecture is flexible across multiple FPGA architectures and the architecture seamlessly adapts to the sizes and speeds of multiple ports without adding to the cost for ports of lower speed or performance for ports of at high speed.

[0042] The present invention provides a multi-processor architecture for a dataplane, which scales across the multiple requirements. It is operated across 5Gbps to 50 Gbps rates as well as three different platforms with different FPGA devices and size. Logic is fully tested for 802.1Q, 802.1ad based switching per MEF standards and MPLS-TP standard as per IETF. This design is verified in 5 different switch designs such as 5G, 6G tested over Lattice ECP3-150 FPGA, 16Gbps on Xilinx Kintex7 FPGA and 36G/48G tested over HXT devices. As the base code is same across all, updates are extremely simple to handle and tests are only incremental.

[0043] Although the embodiments herein are described with various specific embodiments, it will be obvious for a person skilled in the art to

practice the embodiments herein with modifications. However, all such modifications are deemed to be within the scope of the claims.

[0044] It is also to be understood that the following claims are intended to cover all of the generic and specific features of the embodiments described herein and all the statements of the scope of the embodiments which as a matter of language might be said to fall there between.