US 20100135313A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2010/0135313 A1

Davis et al. (43) Pub. Date: Jun. 3, 2010

(54) NETWORK ROUTING SYSTEM FOR ENHANCED EFFICIENCY AND MONITORING CAPABILITY

(75) Inventors: **Ian Edward Davis**, Fremont, CA (US); **Jeffrey A. Prince**, Los Gatos, CA (US); **Ronak Patel**, Santa Clara, CA (US)

Correspondence Address:
Townsend and Townsend and Crew LLP/Foundry/Brocade
Two Embarcadero Center, Eighth Floor
San Francisco, CA 94111 (US)

(73) Assignee: **Foundry Networks, Inc.**, Santa Clara, CA (US)

(21) Appl. No.: **12/624,300**

(22) Filed: **Nov. 23, 2009**

Related U.S. Application Data

(63) Continuation of application No. 10/139,912, filed on May 6, 2002, now Pat. No. 7,649,885.

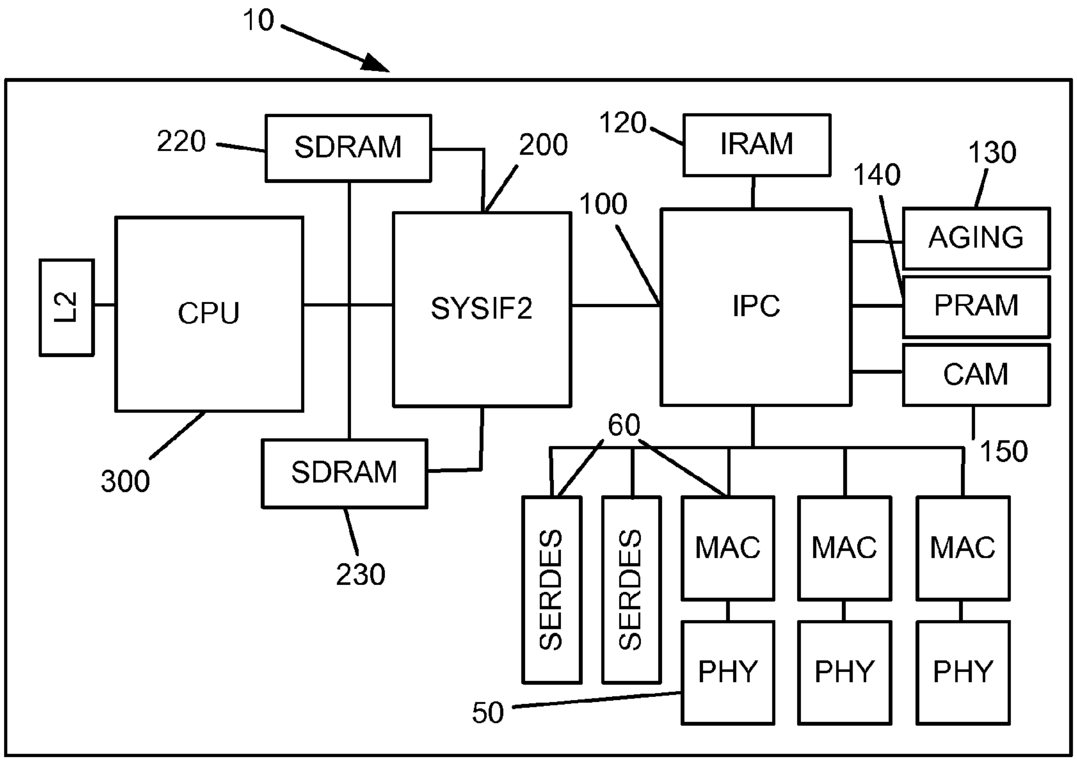
Publication Classification

(51) **Int. Cl.**
H04L 12/56 (2006.01)

(52) **U.S. Cl.** **370/419**

(57) **ABSTRACT**

According to an embodiment of the invention, a network device such as a router or switch provides efficient data packet handling capability. The network device includes one or more input ports for receiving data packets to be routed, as well as one or more output ports for transmitting data packets. The network device includes an integrated port controller integrated circuit for routing packets. The integrated circuit includes an interface circuit, a received packets circuit, a buffer manager circuit for receiving data packets from the received packets circuit and transmitting data packets in one or more buffers and reading data packets from the one or more buffers. The integrated circuit also includes a rate shaper counter for storing credit for a traffic class, so that the integrated circuit can support input and/or output rate shaping. The integrated circuit may be associated with an IRAM, a CAM, a parameter memory configured to hold routing and/or switching parameters, which may be implemented as a PRAM, and an aging RAM, which stores aging information. The aging information may be used by a CPU coupled to the integrated circuit via a system interface circuit to remove entries from the CAM and/or the PRAM when an age count exceeds an age limit threshold for the entries.



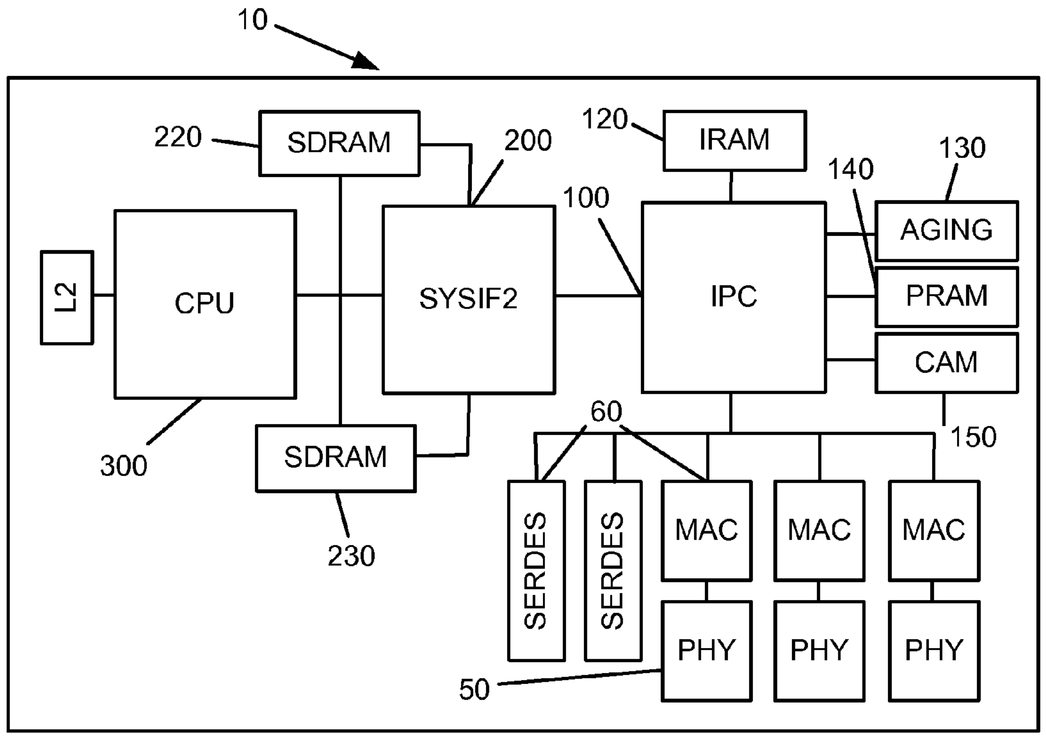

10
220
SDRAM
200
120
IRAM
100
140
130
AGING
L2
CPU
SYSIF2
IPC
PRAM
CAM
60
150
300
SDRAM
230
SERDES
SERDES
MAC
MAC
MAC
PHY
PHY
PHY
50

FIG. 1

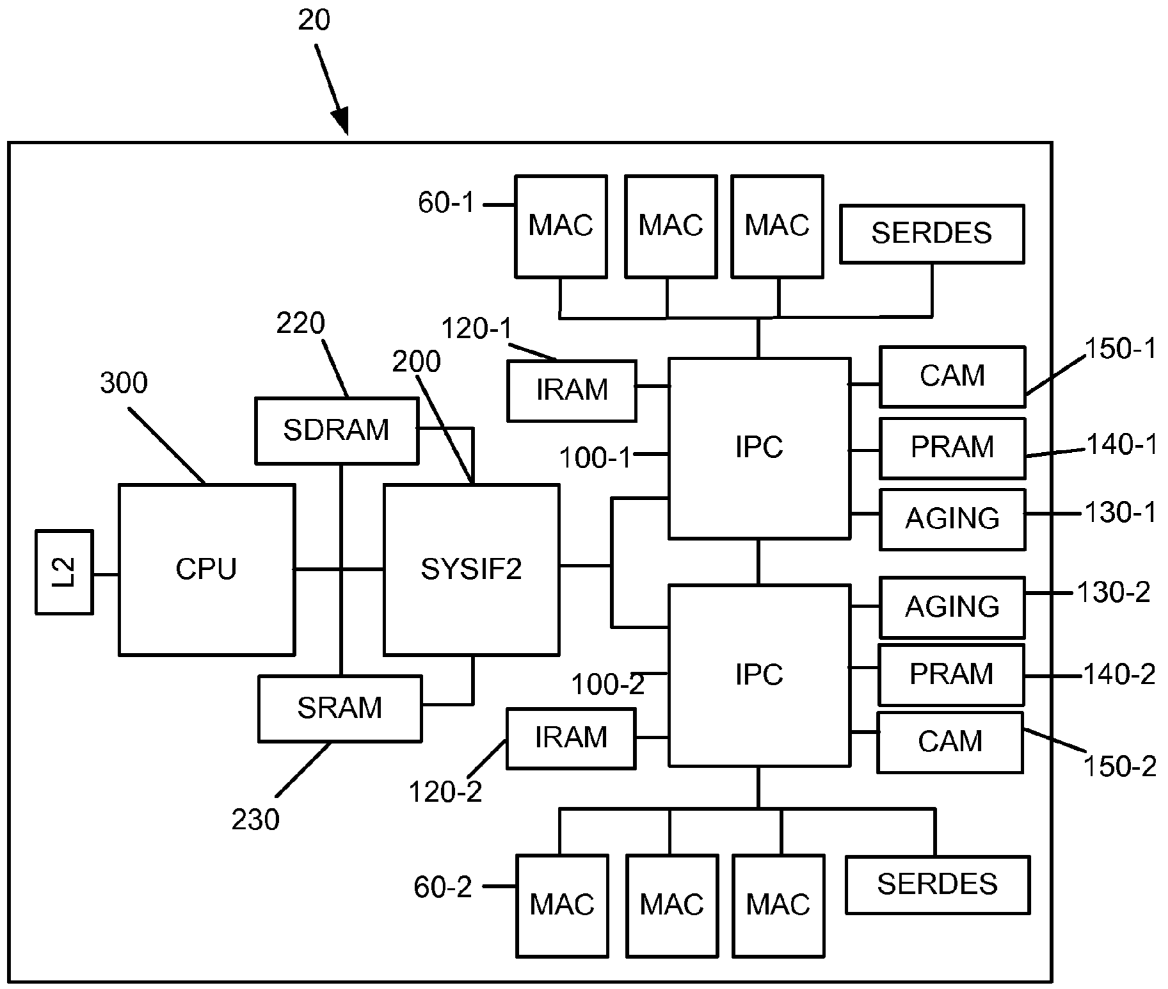

20
60-1
MAC
MAC
MAC
SERDES
220
120-1
200
300
SDRAM
IRAM
CAM
150-1
IPC
PRAM
140-1
100-1
AGING
130-1
L2
CPU
SYSIF2
AGING
130-2
100-2
IPC
PRAM
140-2
SRAM
IRAM
CAM
150-2
230
120-2
60-2
MAC
MAC
MAC
SERDES

FIG. 2

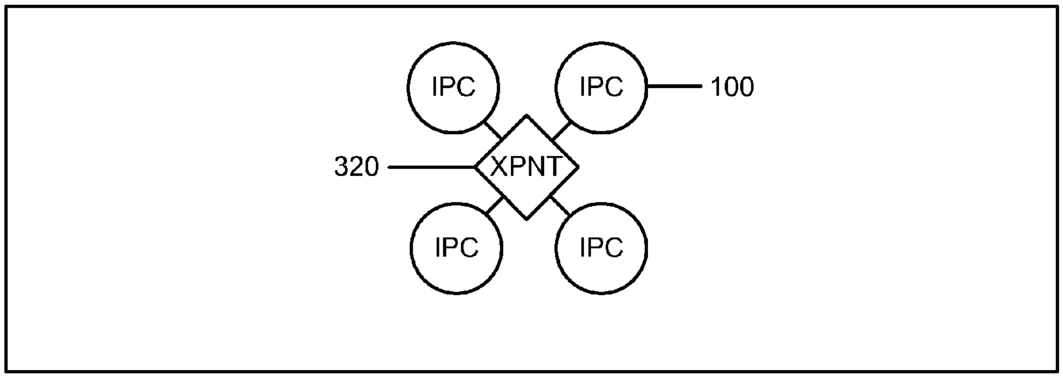


FIG. 3A

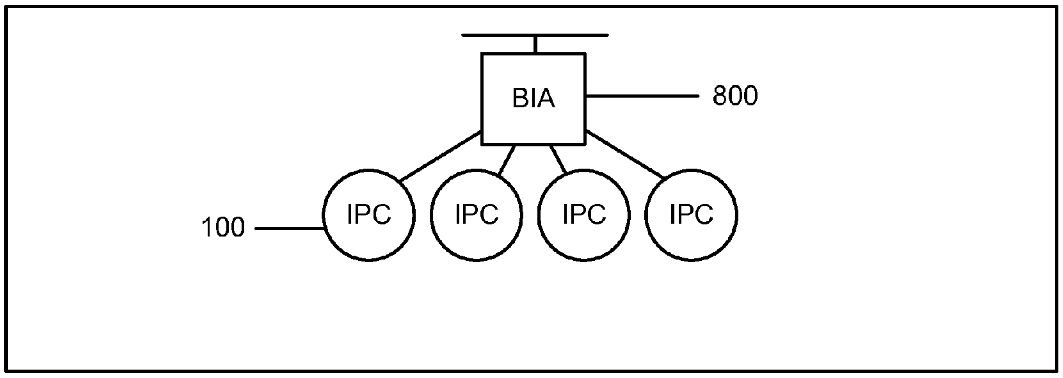


FIG. 3B

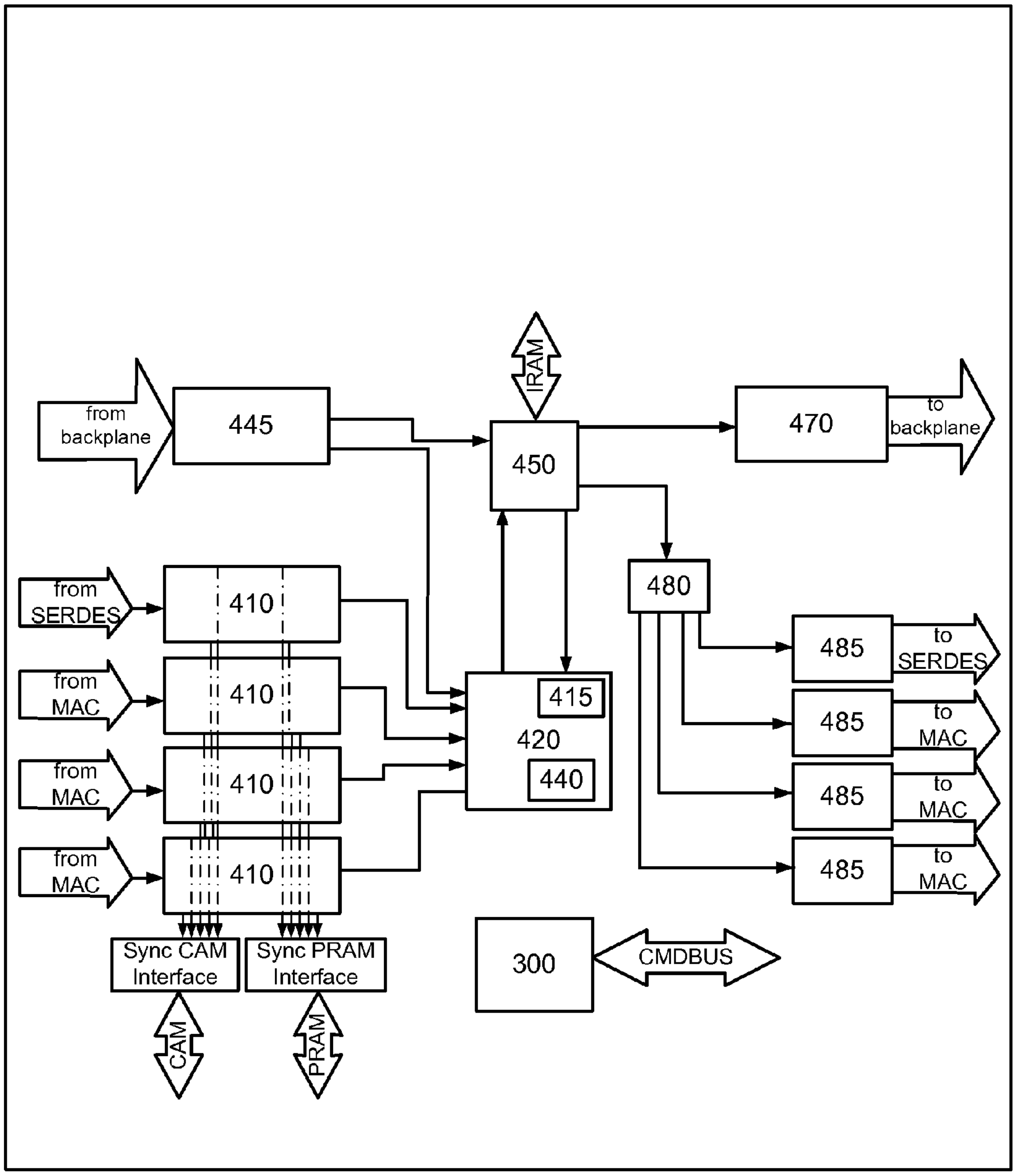

from backplane
445
IRAM
450
470
to backplane
from SERDES
410
from MAC
410
from MAC
410
from MAC
410
415
420
440
480
485
to SERDES
485
to MAC
485
to MAC
485
to MAC
Sync CAM Interface
Sync PRAM Interface
CAM
PRAM
300
CMDBUS

FIG. 4

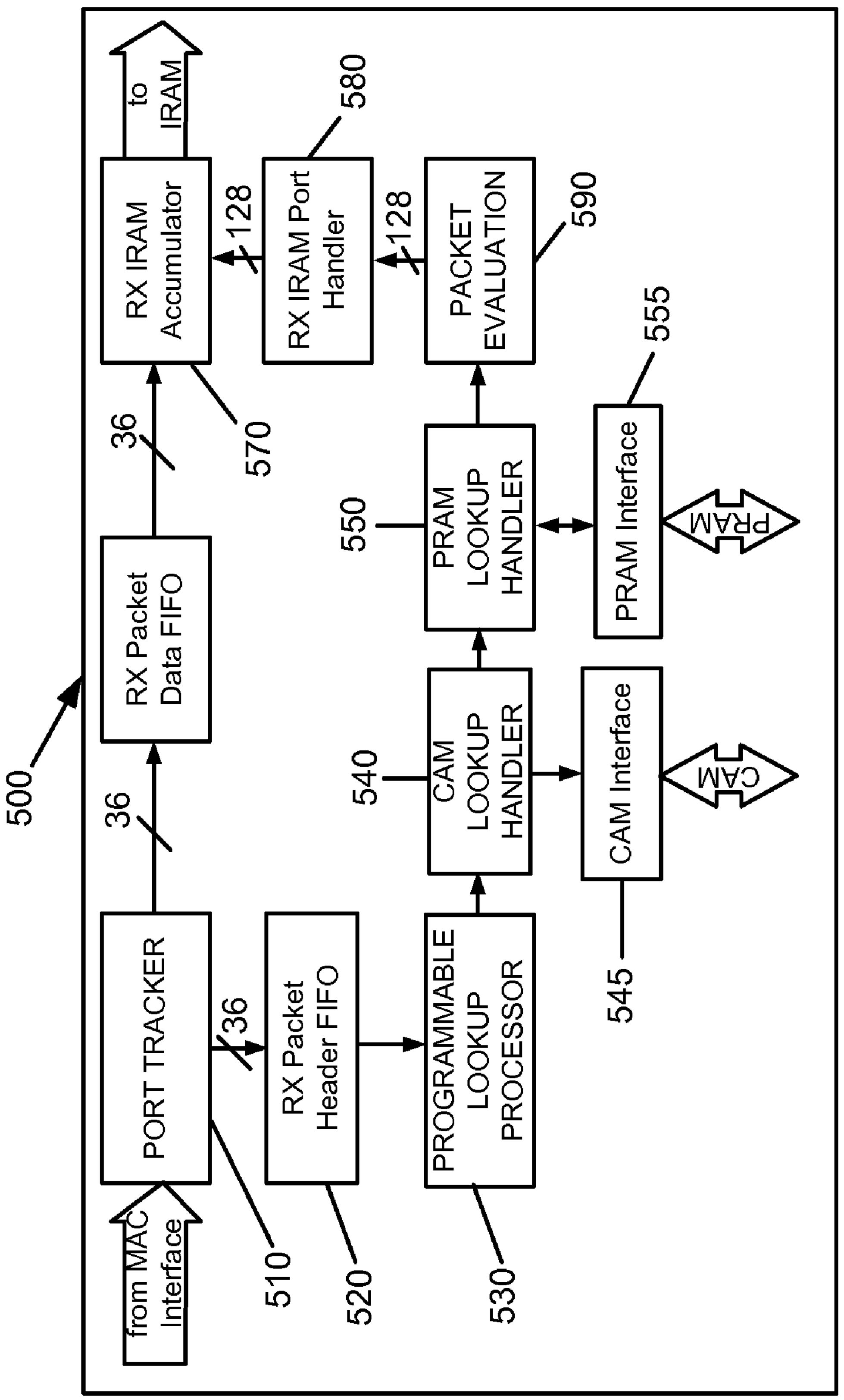

500
from MAC Interface
PORT TRACKER
510
36
RX Packet Header FIFO
520
PROGRAMMABLE LOOKUP PROCESSOR
530
36
RX Packet Data FIFO
36
RX IRAM Accumulator
570
to IRAM
128
RX IRAM Port Handler
580
128
PACKET EVALUATION
590
CAM LOOKUP HANDLER
540
CAM Interface
545
CAM
PRAM LOOKUP HANDLER
550
PRAM Interface
555
PRAM

FIG. 5

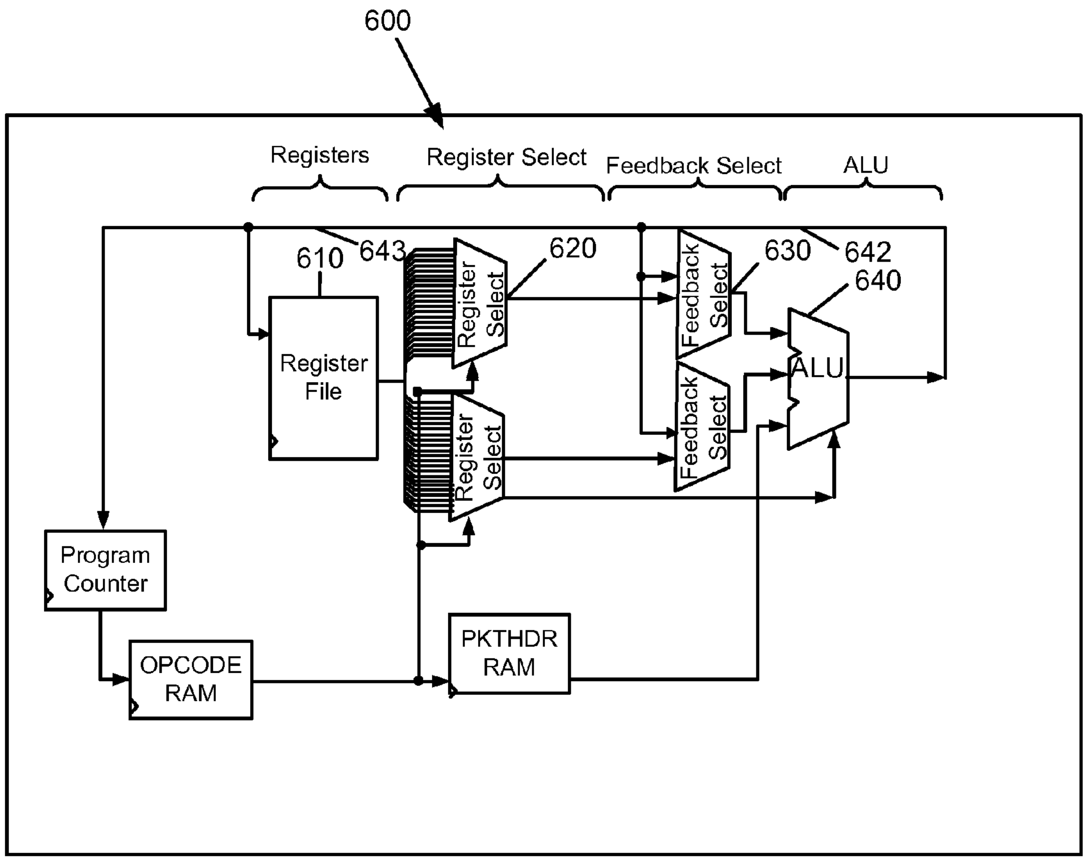

600
Registers
Register Select
Feedback Select
ALU
610
643
620
630
642
640
Register File
Register Select
Register Select
Feedback Select
Feedback Select
ALU
Program Counter
OPCODE RAM
PKTHDR RAM

FIG. 6

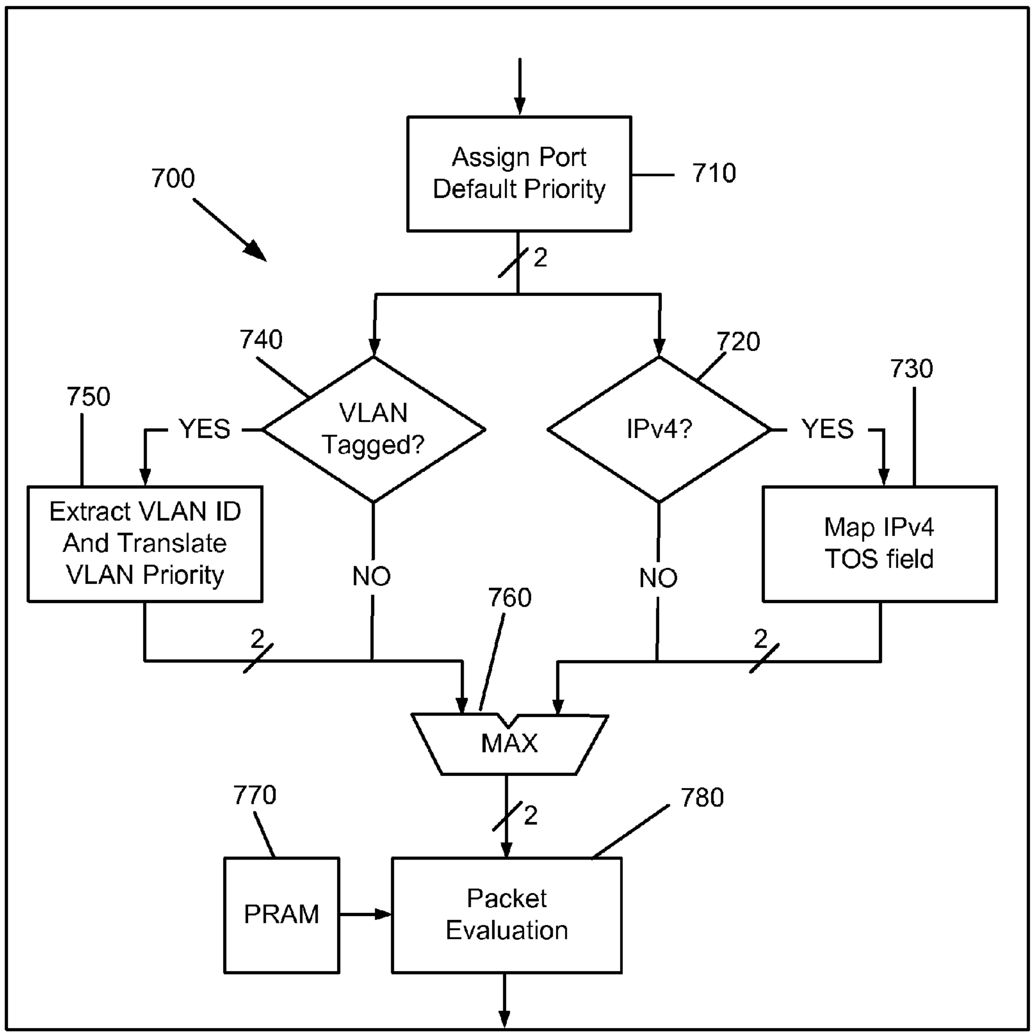

Assign Port Default Priority
710
700
2
740
720
730
750
YES
VLAN Tagged?
IPv4?
YES
Extract VLAN ID And Translate VLAN Priority
Map IPv4 TOS field
NO
NO
760
2
2
MAX
770
2
780
PRAM
Packet Evaluation

FIG. 7

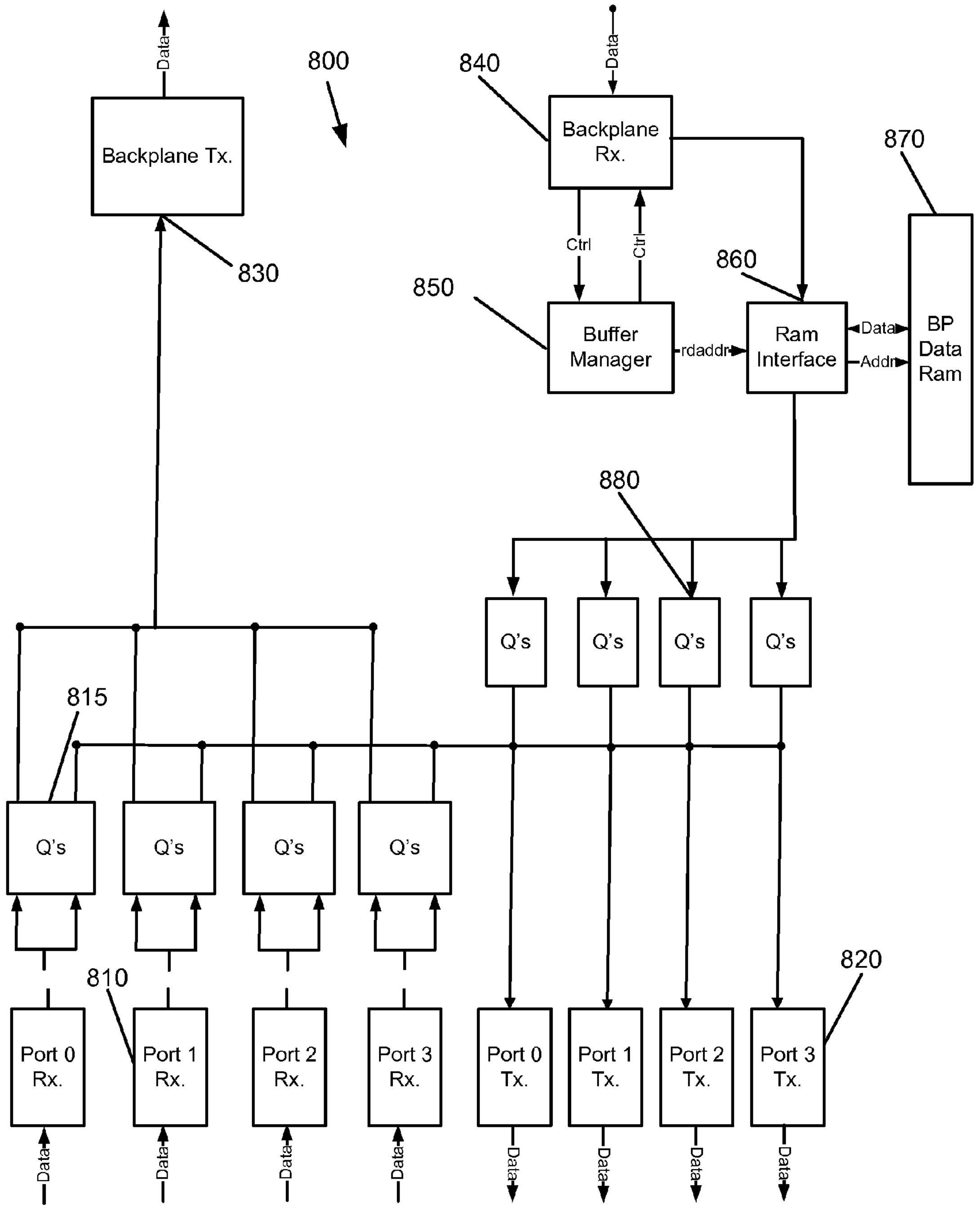

Data
800
840
Data
Backplane Tx.
Backplane Rx.
870
830
Ctrl
Ctrl
860
850
Buffer Manager
rdaddr
Ram Interface
Data
Addr
BP Data Ram
880
Q's
Q's
Q's
Q's
815
Q's
Q's
Q's
Q's
810
820
Port 0 Rx.
Port 1 Rx.
Port 2 Rx.
Port 3 Rx.
Port 0 Tx.
Port 1 Tx.
Port 2 Tx.
Port 3 Tx.
Data
Data
Data
Data
Data
Data
Data
Data

FIG. 8

NETWORK ROUTING SYSTEM FOR ENHANCED EFFICIENCY AND MONITORING CAPABILITY

CROSS-REFERENCES TO RELATED APPLICATIONS

[0001] The present application is a continuation of U.S. application Ser. No. 10/139,912 filed May 6, 2002 and titled "NETWORK ROUTING SYSTEM FOR ENHANCED EFFICIENCY AND MONITORING CAPABILITY" (Atty. Docket No.: 019959-004500US), the entire contents of which are herein incorporated by reference for all purposes.

FIELD OF THE INVENTION

[0002] The present invention relates to routing in a computer network. More particularly, the present invention relates to a system for efficiently routing and monitoring packets in a computer network.

BACKGROUND

[0003] Modern networking environments provide enormously enhanced data transmission capabilities over environments available only a few years ago. However, the demand for bandwidth is constantly increasing, as is the demand for more routing and monitoring capabilities. In order to meet this demand, network devices such as routers need to increase the number of ports serviced and the features they provide.

[0004] For example, network devices need to implement Quality of Service (QOS) features, which can provide better and more predictable network service by ensuring a dedicated bandwidth to be available, improving loss characteristics, avoiding and managing network congestion, shaping network traffic, and setting traffic priorities across the network. Currently, many QOS features are implemented using software. However, software implementation is impractical for the large bandwidth routers needed to handle the increasing amount of network traffic. Similarly, network devices need to be able to route broadcast or multicast packets and jumbo packets, and to provide network monitoring capability.

[0005] Therefore, there is a need for a large bandwidth network device that can efficiently route packets with, for example, "the Internet protocol" (IPv4) type of service (TOS) fields for QOS services. Additionally, the network device should efficiently route jumbo packets and broadcast or multicast packets (including multicast packets with different VLAN IDs). Finally, the network device should be configured to perform network monitoring without the use of additional probes.

SUMMARY

[0006] According to an embodiment of the invention, a network device such as a switch or a router provides large bandwidth as well as efficiency for data packet handling capability. The network device includes multiple input and output ports for receiving and transmitting data packets. According to an embodiment, the network device performs switching or routing of data packets for numerous auto-sensing multi-speed (10/100 megabit) Ethernet ports and very high speed (e.g., gigabit) ports. According to another embodiment, the network device performs switching or routing of data packets for multiple very high speed ports.

[0007] According to one embodiment, the network device provides a port controller integrated circuit for switching or routing packets. The integrated circuit includes a packet input circuit for receiving data packets from at least one of the input ports, and a buffer manager circuit for receiving data packets from the packet input circuitry, transmitting data packets to one or more buffers, and reading data packets from the one or more buffers. The integrated circuit also includes a rate shaper counter for storing credit for a traffic class, so that the integrated circuit can support input and/or output rate shaping.

[0008] The integrated circuit may be implemented as an application specific integrated circuit (ASIC) or in a programmable logic device (e.g., an FPGA). The input ports may be 10/100 megabit Ethernet ports, gigabit Ethernet ports, Packet over SONET (POS) ports, ATM ports, or other ports. The packet input circuitry is configured to provide an interface with the appropriate port type.

[0009] The integrated circuit may be associated with one or more memories which provide a buffer pool for storing data packets. In some embodiments, the buffer pool is implemented using a random access memory (RAM). (The buffer pool is sometimes also referred to as an IRAM.) In other embodiments, other types of memory may be used. The integrated circuit may be associated with one or more content-addressable memories (CAMs) for storing information about the packets ("packet information") being handled in a memory array. The integrated circuit may include a CAM interface used to perform lookups on the CAM.

[0010] In one embodiment, the integrated circuit may be associated with an additional memory provided for storing packet parameters ("PRAM"). Each PRAM stores packet information in a memory array, including switching or routing parameters. The integrated circuit may include a PRAM interface used to perform lookups on the PRAM. The PRAM may be sized to provide values of a predetermined set of packet parameters for each CAM entry.

[0011] The integrated circuit may further include an aging RAM, which stores aging information regarding the CAM and PRAM entries. The aging information may be used by a host CPU, which may be coupled to the integrated circuit via a system interface circuit, to determine for removal entries from either the CAM, the PRAM, or both, when an age count exceeds an age limit threshold for the entries. Age counts are incremented periodically for a CAM entry, unless the entry is referenced, which resets its age count.

[0012] The integrated circuit may include a packet evaluation circuit. The packet evaluation circuit may include a port tracker circuit. The packet evaluation circuit may also include a programmable lookup processor, which may be a RISC processor. The programmable lookup processor may include a register file, a register select circuit for selecting the contents of registers as operands, an arithmetic logic unit for operating on the operands, and a feedback select circuit for providing, alternatively, as operand an output value of the ALU. In one embodiment, the register file is configured such that some of the registers are assigned to particular packet parameters, such that a snapshot of the register file provides without further processing a key for a CAM lookup. The output value of the ALU may be written into one or more of the registers.

[0013] The packet evaluation circuit may also include a CAM lookup handler for submitting lookup requests to the CAM, and a PRAM lookup handler for submitting lookup requests to the PRAM based on the values returned from a CAM lookup. The packet evaluation circuit may include packet evaluation logic circuits for performing packet processing using the results of a CAM lookup and a PRAM lookup.

[0014] The port tracker circuit may identify valid packet contexts (to filter corrupted packet data), copy a VLAN tag to a status word, and remove a VLAN tag from a packet header, in order to facilitate packet processing. The port tracker circuit may also perform TOS field lookups under the IPv4 protocol, or another suitable protocol.

[0015] The packet input circuit may include an 8B/10B decoder. Additionally, the packet input circuit may include logic circuits for CRC verification and auto-negotiation.

[0016] The integrated circuit may further include a polling logic circuit, which may perform time slot polling of the input ports of the network device. The integrated circuit may further include a received data FIFO circuit to receive data packets from the polling logic circuit. The integrated circuit may further include an internal VLAN table.

[0017] The buffer manager circuit may perform rate shaping, including input rate shaping and output rate shaping. The rate shaping may be based on port, both port and priority, or L3/L4 (network level) information. The buffer manager circuit may also be configured to route jumbo packets, which are variable-length packets for very high speed ports.

[0018] A priority may be assigned to a data packet by default, and according to whether the data packet is specified with a VLAN priority or a TOS priority. The packet priority may be further modified from the results of a CAM lookup or a PRAM lookup.

[0019] The processed data packet may be transferred to a buffer in an IRAM by the buffer manager circuit for forwarding. The buffer manager circuit may perform rate shaping. Rate shaping may be achieved by defining traffic classes, and storing credit in a counter corresponding to the traffic class. Credits are added to each counter periodically according to a credit interval. The amount of additional credit added to each counter may be different. The amount of credit is decreased when the buffer manager forwards a packet for the traffic class.

[0020] An interface adapter may be used with a port controller integrated circuit as described above, in order to interface multiple port controller integrated circuits with a backplane having multiple backplane slots. The interface adapter may provide data rate matching where the combined bandwidth of the multiple port controller integrated circuits is different from the bandwidth of the backplane. The interface adapter may transmit packets to and receive packets from any of the backplane slots and any of the port controller integrated circuits. The received data packets and the data packets to be transmitted may be stored in backplane queues. A buffer manager may be provided in the interface adapter for managing buffers used to mediate data packet traffic among the backplane and the port controller integrated circuits. A backplane RAM can be provided to provide buffers for storing data packets in transit among the backplane slots and the port controller integrated circuits.

[0021] A more complete understanding of the present invention and its advantages will be afforded to those skilled in the art upon consideration of the following detailed description of the exemplary embodiments therein. Reference will be made to the appended drawing that will first be described briefly.

BRIEF DESCRIPTION OF THE DRAWING

[0022] FIG. **1** is a block diagram of router **10**, which includes an integrated port controller, according to an embodiment of the invention;

[0023] FIG. **2** is a block diagram of router **20**, which includes two integrated port controllers, according to another embodiment of the invention;

[0024] FIGS. **3**A and **3**B are block diagrams of two configurations in routers where multiple integrated port controllers may be connected, according to other embodiments of the invention;

[0025] FIG. **4** is a block diagram of a port controller ASIC that may be used in a network device, such as the routers of FIGS. **1**, **2**, **3**A, and **3**B, according to an embodiment of the invention;

[0026] FIG. **5** is a block diagram of packet evaluation circuit **500**, suitable for implementation in packet input circuit **410** shown in FIG. **4**, according to an embodiment of the invention;

[0027] FIG. **6** is a block diagram of processor **600**, which is one implementation of PLP **530** of FIG. **5**, according to an embodiment of the invention;

[0028] FIG. **7** shows process steps that may be performed using a router to assign a priority to a packet, according to an embodiment of the invention; and

[0029] FIG. **8** is a block diagram of interface adapter ASIC **800** that may be used in a router such as that shown in FIG. 3B, according to an embodiment of the invention.

[0030] Use of the same or similar reference numbers in different figures indicates the same or like elements.

DETAILED DESCRIPTION

[0031] According to an embodiment of the invention, a network device includes one or more integrated port controllers, each implemented in an integrated circuit such as an application specific integrated circuit (ASIC) or a field programmable gate array (FPGA), to manage and monitor packet flow efficiently.

Network Device with Integrated Port Controller

[0032] In FIG. **1**, a network device such as a router **10** includes a port controller ASIC **100**. Data packets are transmitted to input terminals **60** of port controller ASIC **100** via physical interfaces **50**. Input terminals **60** to ASIC **100** may be provided by media access controller (MAC) circuits, for conventional 10/100 megabit Ethernet ports, or may be provided by serializer/deserializer (SERDES) circuits, for gigabit Ethernet ports. Router **10** may support other types of ports, such as POS ports, ATM ports, or other ports. According to an embodiment of the invention, ASIC **100** has twenty four 10/100 megabit Ethernet ports and two gigabit Ethernet ports. According to an alternate embodiment, ASIC **100** has four gigabit Ethernet ports. In one embodiment, each port is provided a 48-bit MAC address of which the upper 32 bits are common to all the ports, and the remainder 16 bits of the MAC address are programmable.

[0033] ASIC **100** may be associated with one or more memories, such as an integrated packet controller memory ("IRAM") **120**, aging memory **130**, parameter memory (PRAM) **140**, and content addressable memory (CAM) **150**. (Functions of these memories are explained in further detail below). IRAM **120**, aging memory **130** may be implemented by random access memories. Although FIG. **1** shows ASIC **100** to be associated with one memory of each type listed above, in other embodiments more than one memory of a given type may be provided. ASIC **100** is also associated with a system interface chip **200**, which in turn is associated with one or more memories such as memory **220** and **230** of FIG.

1. System interface chip **200** provides an interface between ASIC **100** and a host CPU **300**.

[0034] ASIC **100** may interact with its associated memories as follows. ASIC **100** provides to CAM **150** packet information extracted from a packet received into ASIC **100**, to initiate a search in CAM **150** to determine how to forward the packet to its destination and to initiate other packet processing functions. If a match is found, CAM **150** returns corresponding parameter values; in addition, or alternatively, CAM **150** returns an index into another memory array, where the corresponding data is stored. For example, in a destination address (DA) search, ASIC **100** uses the returned index to retrieve forwarding data from PRAM **140**. For a source address (SA) search, ASIC **100** uses the returned index to retrieve source port information from PRAM **140**, which is then used to age CAM entries.

[0035] PRAM **140** includes additional information for further processing the packet. PRAM **140** may be implemented by a 32-bit synchronous DRAM (SDRAM), sized to match CAM **150**. According to an embodiment of the invention, PRAM **140** includes four separate tables implemented in different SDRAM banks Destination address table records are in one table, source address table records are in another table, L3 (network level) records are in another table, and L4/session (network/session level) records are in another table. This banked table structure permits CAM lookups according to many supported packet types to receive different services at different levels. The associated PRAM data provide destination address/source address lookups and support network monitoring and management functions.

[0036] According to an embodiment of the invention, PRAM **140** implements address aging, which allows a CPU such as CPU **300** of FIG. **1** to remove unused entries from the CAM and PRAM memory arrays. An age bit, including an age count and an age-disable flag, is stored in a PRAM record, as well as in a separate AGERAM record on aging RAM **130**. PRAM **140** also includes an aging configuration register, which may be set with an aging threshold.

[0037] When CAM **150** performs a successful address lookup (that is, locates a matching entry in the CAM array), a PRAM lookup cycle at that CAM index is performed. The information retrieved from PRAM **140** is incorporated into the 16-byte packet status word, and the age count may be zeroed, which is performed after a source address lookup in this embodiment. If the age count is zeroed, it is zeroed both in the PRAM record and the AGERAM record. The aging function is initiated by CPU **300**, which commences an aging cycle by issuing an age cycle command to ASIC **100**. When the age cycle command is received, an aging controller on ASIC **100** scans the AGERAM entries, incrementing the age count whenever the age-disable flag is not set, and the age value is less than an age-limit threshold in the PRAM aging configuration register. An active aging cycle is indicated in the status field of the PRAM control register. PRAM entries that age-out (the age count exceeds the age-limit threshold) have their indices stored in an aging FIFO, so that CPU **300** can take appropriate action; for example, over-writing the CAM and PRAM indices.

[0038] Once all required packet type decoding, CAM, and PRAM lookups are complete, a buffer manager controller such as buffer manager controller **440** of FIG. **4** transfers packets to one or more buffers in IRAM **120**. Buffer manager controller **440** is discussed in further detail below.

[0039] FIG. **2** shows another embodiment of the present invention in router **20**, which includes two port controller integrated circuits to provides support for more input ports and hence a higher traffic level than router **10** illustrated in FIG. **1**. Router **20** includes port controller integrated circuits (i.e., port controller ASICs) **100-1** and **100-2**. ASIC **100-1** and **100-2** are each interfaced to ports (i.e., port input terminals **60-1** and **60-2**) and associated with memories (e.g., IRAM **120-1**, aging ram **130-1**, PRAM **140-1**, and CAM **150-1** are associated with ASIC **100-1**, while IRAM **120-2**, aging ram **130-2**, PRAM **140-2**, and CAM **150-2** are associated with ASIC **100-2**) in the same manner as described above for ASIC **100** of FIG. **1**. Router **20** also includes a system interface chip **200** which provides an interface between CPU **300** and each of ASICs **100-1** and **100-2**.

[0040] FIGS. **3**A and **3**B show other configurations of port controller integrated circuits for network devices capable of handling even greater packet traffic levels. FIG. **3**A shows four port controller integrated circuits such as ASIC **100** of FIG. **1**, coupled via a switch such as a crosspoint switch **320**. FIG. **3**B shows an alternate configuration, where four port controller integrated circuits such as ASIC **100** of FIG. **1** are coupled to a backplane of another router, through an interface adapter integrated circuit. The interface adapter integrated circuit may be implemented as an ASIC such as an interface adapter ASIC **800** of FIG. **3**B.

[0041] FIG. **4** is a block diagram of an embodiment of a port controller integrated circuit such as ASIC **100** of FIG. **1**. As shown in FIG. **4**, ASIC **100** includes packet input circuit **410**, which is configured to interface with gigabit ethernet media access channel (GMAC) ports and may contain an 8B/10B encoder/decoder and logic circuits for CRC verification, and auto-negotiation. In addition, packet input circuit **410** may be configured to interface with conventional 10/100 Ethernet media access controller (MAC) ports. Packet input circuit **410** may additionally receive packet transfers and perform time-slotting of transmit packet transfers. For example, packet input circuit **410** may receive packet transfers in bursts of sixteen cycles. In some embodiments, packet input circuit **410** may be configured to interface with other ports such as ATM ports or POS ports, or may be a combination of different interface types.

[0042] Besides forwarding packets to their destinations, packet input circuit **410** performs further functions. Packet input circuit **410** may be configured to perform packet classification, prepare packet modifications, and generate packet headers, which are functions that can be used to support routing at higher protocol levels, network traffic management and monitoring. Further, packet input circuit **410** prepares sixteen-byte encapsulation, which used in forwarding packets through router **10**. In FIG. **4**, packet input circuit **410** is implemented in separate blocks for each input port. According to other embodiments, a single block may provide input circuitry for a single input port, or for more than one input port. Input circuitry **410** may be different for different input port types, or only a sub-unit of input circuitry **410** may be different.

[0043] FIG. **5** is a block diagram of packet evaluation circuit **500**, which may be included in packet input circuit **410** or elsewhere on ASIC **100**. As shown in FIG. **5**, a received packet is received into port tracker **510**. Port tracker **510** performs "sanity" checks on the data packet received into ASIC **100** through, for example, one of the MAC interfaces, such as identifying valid packet contexts (e.g., consistent start of packet and end of packet boundaries) and examining the status word appended by the MAC, which indicates any data faults. In addition, port tracker **510** strips virtual local area network (VLAN) tags, and places a copy of the first 60 bytes of packets into header first-in-first-out (FIFO) memory **520**, and a copy of the entire packet into packet data FIFO memory **560**. Port tracker **510** may also perform some basic packet decoding, such as comparing the packet MAC destination address (DA) against the port MAC address, and checking the Ethernet Type field to determine whether the received packet has a VLAN tag. If DA matches the port MAC address, an internal status bit ("RX_US") is set. Based on this internal status bit, a data packet having a DA in ASIC **100** is routed to CPU **300**. According to an embodiment, the VLAN ethertype field is fully programmable. When a received packet has a VLAN tag, the VLAN tag is copied from the header into a 16-byte packet status word, then removed from the packet header, so that packet processing in some portions of packet evaluation circuit **500** can proceed without regard to whether the packet is associated with a VLAN. For IPv4 type packets, port tracker **510** may also perform TOS field lookups, to enable input and output rate shaping (see below). The results of all evaluations are placed into bytes **60-63** of the packet header data.

[0044] Received packet headers are forwarded to received packet header FIFO memory **520**. In an embodiment, received packet header FIFO memory **520** has a capacity of 256×36 bits. Received packet data is forwarded to a received packet data FIFO memory **560**. According to an embodiment, received packet data FIFO memory **560** has a capacity of 256×36 bits.

[0045] Packet header data is forwarded from received packet header FIFO memory **520** to a programmable lookup processor (PLP) **530** for further processing. PLP **530** forms CAM lookups, creates part of the 16-byte packet header for the outgoing packet to be forwarded, and generates information needed for packet evaluation to function properly. Based on packet type (e.g., IP, IPX or L2), PLP **530** also computes a trunk index to support trunking This trunk index is used to logically 'ORed with a MAC destination address FID.

[0046] In one embodiment, PLP **530** is a 16-bit RISC processor, able to access anything from the first 60 bytes of a packet. A program drives the specific operations of PLP **530**, which directs the types of CAM lookups to be carried out, according to the packet type and values of system parameters. Some registers in the RISC processor of that embodiment are assigned to specific parameters that comprise the packet context, so that their contents can directly compose specific L2/L3/L4 CAM targets or contain packet header fields. Once processing is complete the packet context is transferred to the CAM lookup handler **540**.

[0047] FIG. **6** is a block diagram of processor **600**, which is one implementation of PLP **530** of FIG. **5**. Processor **600** includes register file **610**, register select block **620**, feedback select block **630**, and arithmetic logic unit (ALU) **640**. In an embodiment, register file **610** includes thirty one general purpose 16-bit registers and one program counter register. The registers can be freely used during evaluation to perform any operation. However, once evaluation is complete the register contents can be directly used for CAM targets and packet header information.

[0048] Register select block **620** chooses a target register's contents from register file **610** as operands into ALU **650**. Feedback select block **630**, which selects either the operands from the register select block **620**, or an output value of ALU **620**, permits back-to-back use of modified registers. In this implementation, the registers in register file **610** are pipelined such that a write operation into a register in register file **610** takes two processor clock cycles. However, if processor **640** detects that a result from ALU **640** is used in the following instruction, feedback select block **630** selects the result from ALU **640** as operand for this following instruction, rather than from register file **610**. ALU **640** supports load and store operations, and arithmetic and logic binary operators including and, or, xor, neg, add, compare, inline rotate and mask operations. Constants, or immediates, can be substituted for register values in places.

[0049] Once PLP **530** completes its operation, the contents of register file **610** are transferred to CAM lookup handler **540**. CAM lookup handler **540** takes a snapshot copy of all the PLP registers and submits these values to initiate one or more CAM look-up requests via CAM interface **545**. With CAM lookup handler **540** controlling CAM lookup operations, PLP **530** can begin to work on another packet. When the CAM returns the lookup results, the context is transferred to a PRAM lookup handler **550**.

[0050] Like CAM lookup handler **540**, PRAM lookup handler **550** is also a placeholder. Specifically, PRAM lookup handler **550** maintains the packet contexts while PRAM lookups are performed. CAM handler **540** and PRAM lookup handler **550** allow a pipelined operation in the units of packet evaluation circuit **500**, so that useful work (instead of stalling) is carried out while the memory accesses (e.g., such as PRAM data transfers) are performed. PRAM lookups are submitted to the PRAM via PRAM interface **555**. After PRAM lookups are complete, further packet processing may be performed in packet evaluation block **590**.

[0051] In most packet types, CAM lookups are carried out for the destination address and the source address. Additional lookups may be carried out for some packet types. For example, if the packet type is IPv4 or IPX, another CAM lookup (for level 3, or network layer routing information) may be done. If the packet type is IPv4, a level 4 or session lookup may also be carried out. After a successful CAM lookup, a PRAM lookup may be performed to obtain additional information used in packet forwarding. During the CAM and PRAM lookups, a number of status word flags may be set up, as an aid to software packet forwarding, hardware packet forwarding, or both. For some packet forwarding, the destination address may be replaced, or the packet header may be modified, or both in order to support hardware packet routing.

[0052] Received IRAM port handler **580** transfers data in received packet data FIFO **560** to received IRAM accumulator block **570**, which is then provided to IRAM **120** (FIG. **1**).

[0053] In one embodiment, a separate IRAM port handler handles packets for each of ports **60**. According to one embodiment, IRAM accumulator block **570** handles read data from port receive FIFOs in 32 byte chunks, applying packet modifications, and dumping data into an IRAM received FIFO. It also detects the end of packet, and builds RXDONE messages for buffer manager controller **440** of FIG. **4** (described in further detail below). If a packet is flagged as bad (for example, due to an invalid CRC), buffer manager controller **440** re-circulates the buffer directly into a freelist.

[0054] Referring again to FIG. **4**, received packets are forwarded from packet input circuit **410** to packet routing circuit

420. In one embodiment, packet routing circuit **420** may includes a packet polling circuit, which performs time slot polling of the input ports for received packet data. In FIG. **4**, the packet polling circuit is included in packet polling logic block **415**, which is shown as part of packet routing circuit **420**. In other embodiments, the packet polling logic circuit may be located differently on ASIC **100**. In one embodiment, packet data is accumulated into 128 bit words and forwarded by packet routing circuit **420** to a buffer pool in IRAM **120** of FIG. **1**, after all appropriate packet modifications are performed packet input circuit **410**, packet evaluation circuit **500** described above, or elsewhere on ASIC **100**. Packet routing circuit **420** obtains and assigns buffer numbers, in specify where to store packets, and to inform buffer manager controller **440** how to forward the packet. Buffers assigned to bad or aborted packets are reused.

[0055] In one embodiment, packet routing circuit **420** implements queue management using, for example, FIFO memories. For example, a FIFO memory may be configured to store data subsequent to the packet polling logic circuit, and to provide an asynchronous boundary between received packet processing in packet routing circuit **420** and IRAM **450** of IRAM **420** (FIG. **4**). Further, a FIFO memory may be used to transfer forwarding identifier (FID) and buffer number (priority and source port) information to buffer manager circuit **440** or elsewhere, to enable transmit queuing.

[0056] Buffer manager controller **440** handles transmit port queuing and rate shaping of the packet data streams. In one embodiment, buffer manager controller **440** receives RXDONE messages from port and backplane logic blocks, each indicating a complete packet evaluation. Buffer manager controller **440** extracts the packet's forwarding identifier (FID) and requests a lookup from IRAM interface **450**. IRAM interface **450** may be separate from packet routing circuit **420** or may be implemented elsewhere in the switch or router. In some embodiments, buffer manager controller **440** is configured to perform source port suppression or to merge CPU and monitor masks. Buffer manager controller **440** may then add packets to individual port queues at, for example, 22 milion packets per second (Mpps). In some embodiments, buffer manager controller **440** also directs port transmit activity. For example, buffer manager controller **440** may explicitly informs IRAM interface **450** to send packets in a particular buffer pool data to particular ports, such as ports **485** of FIG. **4**, or backplane slots, such as slots **470** of FIG. **4**. Once packets are fully dispatched, the buffers are returned to the packet freelist.

[0057] In some embodiments, buffer manager controller **440** may support input rate shaping. Input rate shaping allows for a large number of different traffic classes to be defined and independently controlled based on programmable bandwidth limits. For example, Table 1 shows three modes of operation for an embodiment incorporating input rate shaping.

TABLE 1

Mode	Description
Port based	Port based is the most basic form of input rate shaping. In this mode, each port's receive data is mapped to a traffic class, and each port's class can be independently controlled
Port and priority based	Port and priority based input rate shaping uses both the source port number and the packet priority to create a traffic class. In an embodiment, each port can have up to four traffic classes within it, and each can be independently controlled.
L3/L4 info based	L3/L4 info based input rate shaping uses a field in the PRAM (TOS replacement field) to allow software to define traffic classes based on packet IP/IPX addresses. Because the TOS field is used, this operation is only allowed in Layer 3 and Layer 4 modes of operation, and the TOS replacement cannot be used when using this mode.

[0058] A number of parameters I, V, C, B and T are used to configure and control the input rate shaping for each class. Interval time I is the amount of time between the adding of credits for each traffic (rate shape) class. According to one embodiment of the invention, a single interval time applies to all traffic classes. In that embodiment, the selected interval period spans the entire range of traffic patterns to shape. In one embodiment, a maximum value of the interval time may be 19.66 ms, while a minimum value, which may be a default, may be chosen as 19.2 μs. Credit value V equals to the number of bytes each credit represents. According to one embodiment of the invention, a single credit value applies to all traffic classes and may have values ranging from 32 to 256 bytes per credit, in powers of 2. Credit per interval C is the amount of credit to give at the end of each interval time. Credit per interval C may be programmed to be different for each traffic class. Credits may be added to a class in two ways: fixed mode, where the programmed credit is stored in a rate shaper counter which is decremented as packets arrive, or accumulate mode, where the programmed credit is added to any credit that was left over from the previous interval. According to an embodiment of the invention, credit per interval C may range from 0 to 4096 in powers of 2. Maximum burst B sets the maximum number of credits that can be accumulated for a port operating in the accumulate mode described above. In effect, it sets a maximum burst value when a port goes from idle to sending packets. According to one embodiment of the invention, the maximum burst may be programmed individually for each traffic class and may range from 0 to 4096 in powers of 2. Credit total T is a counter per port which keeps track of the current amount of credit the port has for packets to pass and, in one embodiment, may range from 0 to 4096 in powers of 2.

[0059] According to an embodiment, at the end of each interval time I, the input rate shaper scans through all 128 traffic classes and either add (accumulate mode) or store (fixed mode) programmed credit C into a counter for each class. Total credit T in the counter cannot exceed maximum burst B. As packets arrive for a given class, the input rate shaper divides the packet length by credit value V, deducts the quotient from total credit T in the counter for that class—if total credit T is greater than the quotient—and allows the packet to be forwarded. Otherwise, the packet is dropped and not counted.

[0060] According to some embodiments, buffer manager controller **440** may support output rate shaping in a similar fashion.

[0061] In one embodiment, IRAM interface block **450**, which accepts data transfer requests from six sources and performs data transfers using a time slot driven rotation, provides access to a wide high bandwidth memory pool in IRAM **120**. The six sources are, respectively, (1) a port received packet path request, where data and address are provided by a port received block; (2) a backplane received packet path request, where data and address are provided by the backplane received block; (3) a buffer manager circuitry FID lookup, where a target FID is provided by the buffer manager circuitry; (4) a buffer manager controller port transmission request, where the buffer pool address and destination backplane slot are provided by the buffer manager circuitry; (5) a CPU read, where the buffer pool address is provided by a command bus interface, and (6) a CPU write request, where the data and address are provided by a command bus interface. CPU operations over a command bus interface may be pipelined.

[0062] Backplane receive interface circuitry **445** receives packets from the backplane and routes them to IRAM interface **450** and packet routing circuit **420**.

[0063] The processing of transmit packets is simpler than that of received packets, since there are no CAM or PRAM lookups to perform. According to an embodiment of the invention, transmit packet processing circuit **480** of FIG. **4** requests data from buffer manager controller **440** when sufficient space is available in the transmit FIFO for a given port. When a packet is available, the integrated packet controller transfers a block of data from IRAM **120** of FIG. **1**. The contents of the packet status word header direct operation of the transmit logic circuit. Transmit packet processing circuit **480** examines the packet header of each packet to determine the packet's length, starting offset, and the type of packet processing needed. Processing depends on the status bits in the header and the port's mode of operation, and includes, for example, dynamically extending or shrinking packet data length and re-aligning data to a quad-word (i.e., 64-bit) boundary. If the packet is VLAN-tagged (see below), processing includes inserting a VLAN ID from the header into the packet (if in auto or tagged mode of operation). Other processing, such as replacing the MAC destination address in packet data with a value from the header and replacing the MAC source address in packet data with port address, are also carried out when required.

[0064] Once the packet header has been processed it is passed to transmit interface circuit **485**. Transmit interface circuit **485** may be a MAC interface controller for transmission to an external MAC. Packets may be transmitted to a backplane of a switch or a router via backplane transmit interface circuit **470** (FIG. **4**).

[0065] VLAN Tagging Support

[0066] According to some embodiments, an integrated port controller such as ASIC **100** of FIG. **1** supports VLAN tagging. In one embodiment, a number of VLAN tagging modes are supported: (1) tagged only ports; (2) untagged only ports, (3) priority tagged only ports, (4) repeater mode auto-tagging ports (tag if necessary), (5) untagged to tagged translator mode (tagging preferred) auto-tagging ports, (6) priority-tagged to tagged translator mode (tagging preferred) auto-tagging ports; (7) and untagged to priority-tagged translator mode (priority-tagging preferred) auto-tagging ports.

[0067] Internal VLAN table

[0068] According to some embodiments, ASIC **100** has an internal VLAN table. L2 VLAN lookups are performed from the internal table. The VLAN lookup can override, for example, the default FID, the QOS (Quality of Service) index, and enforce per-port VLAN blocking.

[0069] Packet Priority Handling

[0070] A network device such as router **10** of FIG. **1** may allow for different forwarding priorities of data packets. Packet forwarding priority within router **10** may be established in a number of ways. Packet priority may be based on packet evaluation parameters, such as those determined during CAM and PRAM lookups. Additionally, priority may be affected by VLAN tags and TOS (type of service) lookups.

[0071] FIG. **7** shows a process **700** for assigning packet forwarding priority, according to one embodiment of the invention. In step **710**, a 2-bit port default priority is assigned to a packet. In step **720**, the packet's packet type modifies its packet forwarding priority. If the packet type is IPv4, the IPv4 TOS field replaces the port default priority. Alternatively, a VLAN tag also modifies the packet forwarding priority, as shown in step **740**. If a packet has a VLAN tag, its VLAN ID is extracted in step **750**, and a VLAN priority is translated and replaces the port default priority.

[0072] In step **760**, the highest of the applicable priorities is selected. The highest priority may be the port default priority, the VLAN priority, or the priority in the TOS field.

[0073] In step **770**, the PRAM produces a 3-bit merge value. In step **780**, a resulting packet priority is determined from the 3-bit merge value and the 2-bit priority from step **760**. Table 2 below lists the results obtained for different merge values.

TABLE 2

Merge Value	Result
000	Max of (2-bit priority, 0)
001	Max of (2-bit priority, 1)
010	Max of (2-bit priority, 2)
011	Max of (2-bit priority, 3)
100	Force to 0
101	Force to 1
110	Force to 2
111	Force to 3

[0074] Jumbo Packet Support

[0075] According to an embodiment of the invention, a network device such as router **10** of FIG. **1** may support jumbo packet sizes. To route jumbo packets, a buffer size (e.g., up to 15 Kybtes or higher) is set in IRAM **120** to accommodate jumbo packets. Additionally, GMAC ports or back plane slots capable of sending or receiving jumbo frames are identified and enabled. Buffer manager controller **440** may be configured to enable forwarding jumbo packets to 10/100 Mbit Ethernet ports. Additionally, buffer manager controller **440** may be configured to copy a jumbo packet to the CPU if a destination is dropped because it cannot handle jumbo frames.

[0076] Multicast Packet Support

[0077] A network device such as router **10** of FIG. **1** may also support broadcast or multicast packets (i.e, a received packet is replicated multiple times and transmitted to designated ports). Multicast packets may be transmitted with different VLAN IDs. By setting a flag in the packet header, buffer manager controller **440** recognizes the packet as a multicast packet with VLAN replication enabled. The VLAN ID in the packet header is then treated as a multicast VLAN identifier (MID), enabling packet replication with the correct VLAN ID. In one embodiment, the MID and a transmit port number are used to compute an index into a "multicast start offset table" to obtain a replication count for the transmit port. In this manner, the multicast can be treated differently for each port. The count for each transmit port is used to index into a multicast replacement table. As the count is incremented for each replication, the count points to a different replacement table record in the multicast replacement table. The replacement record provides the VLAN ID to use, the VLAN priority to use and other special instructions for processing the replication.

[0078] Trunking Support

[0079] In addition to the FID adjustment based on packet address and packet type, FID adjustment to support trunking can also be based on the physical port number. In one embodiment, selected bits (e.g., bits [4:1]) of the physical port number can be used to modify the FID by an logical 'OR. Alternatively, masked source port suppression on a per-port basis allows portions of the port number to be ignored during segment filtering. Packets arriving from any of the trunked ports segment filters to the same destination.

[0080] Statistical Packet Sampling

[0081] A network device such as router **10** of FIG. **1** may be configured to perform statistical packet sampling to monitor and analyze network traffic. A commonly assigned U.S. patent application entitled "Network Monitoring Using Statistical Packet Sampling," Sunil P. Chitnis, Ian E. Davis, Jordi Moncada-Elias, Satyanarayana M. Sama, attorney docket number M-12611 US, filed on Mar. 26, 2002, which is hereby incorporated by reference in its entirety, describes statistical packet sampling in a network device such as router **10**.

Interface Adapter

[0082] According to some embodiments, an integrated port controller such as ASIC **100** described above may be used with an interface adapter (IA), which is implemented in an integrated circuit such as an ASIC **800** shown in FIG. **8**. ASIC **800** may provide an interface between one or more integrated port controllers and a backplane, as shown in FIG. 3B. For example, ASIC **800** may provide an interface between four integrated port controllers and seven backplane slots.

[0083] An interface adapter such as ASIC **800** may be used to transmit data when more than one integrated port controller such as ASIC **100** of FIG. **1** is configured to route data to and from a backplane on a network device such as router **10**. The interface adapter can manage bandwidth difference between multiple port controllers such as ASIC **100** and the backplane of the network device. By providing an interface adapter such as ASIC **800**, a higher density per line card may be achieved.

[0084] Integrated Port Controller Receive Interface Block

[0085] An integrated port controller receive interface block **810** interfaces with an integrated port controller such as ASIC **100** of FIG. **4**. According to an embodiment, block **810** receives data from an integrated port controller on a 32-bit data bus. Block **810** also receives a 3-bit header, and a destination port number. The destination port number specifies which of the ports or backplane slot the 32-bit data should be sent. Data packets received in block **810** can be transmitted to one or more backplane queues **815**. Backplane queues **815** transmit data packets to a backplane transfer interface block **830**.

[0086] Integrated Port Controller Transmit Interface Block

[0087] Similarly, an integrated port controller transmit interface block **820** interfaces with an integrated port controller such as ASIC **100** of FIG. **4**. According to an embodiment, block **820** transmits data to an integrated port controller using a 32-bit data bus. Block **820** also transmits a 3-bit header, and a source port number. The source port number specifies which of the ports or backplane slot the 32-bit data originated.

[0088] Backplane Transmit Interface Block

[0089] Backplane transmit interface block **830** interfaces with a backplane on a network device such as router **10** of FIG. **1**. According to an embodiment, block **830** transmits data to the backplane using a 64-bit data bus. Block **830** also transmits a 6-bit header, and a 3-bit slot number that identifies the destination slot for the data.

[0090] Backplane Receive Interface Block

[0091] Similarly, a backplane receive interface block **840** interfaces with a backplane on a network device such as router **10** of FIG. **1**. According to an embodiment, block **840** receives data from the backplane using a 64-bit data bus. Block **840** also receives a 6-bit header, and a 3-bit slot number that identifies the source of the data.

[0092] Buffer Manager

[0093] Interface adapter **800** includes a buffer manager **850**. Buffer manager **850** manages one or more buffers, which receive incoming data from the backplane. According to an embodiment, buffer manager **850** manages buffers that are 256 bytes wide and support 512 KB of data.

[0094] Buffers are allocated using a free buffer list. According to an embodiment, the free buffer list is a 2048-entry circular queue initialized by software during a software reset initialization. Buffer manager **850** allocates a new buffer when the start of a packet is detected from any backplane slot, and when the first bytes arrive from a slot needing another buffer to accommodate the remaining portion of the packet. When a buffer is full, or an end of packet is detected, the header queues corresponding to that packet are updated, as is information in the usage buffer. According to an embodiment, the usage buffer is 2K by 4 bits, where the 4 bits each correspond to an integrated port controller that the buffer contents may be sent to. When the header queue is updated, the buffer entry in the usage buffer is updated with information from an FID RAM, indicating which integrated port controller the buffer contents will be sent to.

[0095] Buffer manager **850** controls the header queues. According to an embodiment, there are 28 header queues, each corresponding to a combination including one of seven backplane source slots and one of four integrated port controllers. Each of the 28 header queues contains 1024 entries. When a header queue fills up, buffer manager **850** sends a hold request to the corresponding backplane slot. A header queue entry is updated when a buffer fills up or when an end of packet is detected.

[0096] Backplane RAM Control Interface Block and Backplane Data Ram

[0097] According to an embodiment, a backplane RAM control interface block **860** provides an interface to a backplane data RAM **870**. Data arrives from the backplane during each cycle. Backplane receive interface block **840** packs two 64-bit data blocks to form a line, which is written to backplane data RAM **870**. The data, as well as an address, are sent to backplane data RAM **870**. According to an embodiment, this write request is considered the highest request and the controller guarantees that the request is honored every time. A FIFO is not used between backplane receive interface block **840** and backplane RAM control interface block **860**, since the write requests are always honored and never delayed or dropped. Data received from the backplane is stored in one or more backplane queues **880**.

[0098] Backplane RAM control interface block **860** is also responsible for interfacing with the read queues which contain addresses from which to read data and place in queues going to integrated port controller transmit interface blocks **820**. Buffer manager **850** provides source slot number and header information corresponding to the data to be read from integrated port controller transmit interface block **820** to the backplane RAM control interface block **860**. Unlike write requests, read requests are arbitrated in a round-robin scheme. When no data is being sent from the backplane, all of the bandwidth is available to process read requests.

[0099] CPU Interface

[0100] Interface adapter **800** may interface with a CPU such as CPU **300** of FIG. **1** via a command bus, which may be a purely asynchronous bus.

[0101] While particular embodiments of the present invention have been shown and described, it will be obvious to those skilled in the art that changes and modifications may be made without departing from this invention in its broader aspects and, therefore, the appended claims are to encompass within their scope all such changes and modifications as fall within the true spirit and scope of this invention.

1. A network device, comprising:
- a plurality of input ports each configured to receive incoming data packets;
- a central processing unit; and
- an integrated port controller integrated circuit, under control of the central processing unit, configured to route the incoming data packets as outgoing data packets; and
- a plurality of output ports, configured to transmit the outgoing data packets.

2. A network device, as in claim **1**, wherein the integrated port controller integrated circuit comprises:
- a packet input circuit, configured to determine the destination of the data packets and to prepare associated header information for preparing outgoing data packets; and
- a buffer manager circuit, configured to store and retrieve contents of the data packets and the header associated information into and out of a plurality of buffers; and
- a packet output circuit, configured to assemble the outgoing data packets from the contents of the data packets and the associated header information.

3. A network device, as in claim **2**, wherein the integrated port controller integrated circuit includes a rate shaper.

4. A network device as in claim **1**, further comprising a second integrated port controller integrated circuit coupled to the first integrated controller integrated circuit and under control of the central processing unit, configured to route the incoming data packets as outgoing data packets.

5. A network device as in claim **4**, wherein the first and second integrated port controller integrated circuits are coupled through a cross-point switch.

6. A network device as in claim **4**, further comprising a back plane interface adapter coupling the first and second integrated port controller integrated circuits to a backplane of the network device.

7. A network device as in claim **1**, wherein the integrated circuit comprises an ASIC.

8. A network device as in claim **1**, wherein the integrated circuit comprises a field programmable gate array.

9. A network device as in claim **1**, selected from the group consisting of a router and a switch.

10. A network device as in claim **1**, wherein the input ports comprise one or more gigabit Ethernet ports.

11. A network device as in claim **1**, wherein the input ports comprise one or more 10/100 megabit Ethernet ports.

12. A network device as in claim **1**, wherein the input ports comprise one or more packet over SONET ports.

13. A network device as in claim **1**, wherein the input ports comprise one or more ATM ports.

14. A network device as in claim **1**, further comprising a memory providing buffers for data packets.

15. A network device as in claim **1**, further including a content addressable memory (CAM) coupled to the integrated circuit, said CAM storing processed data packet information.

16. A network device as in claim **1**, further comprising a system interface circuit coupling the central processing unit to the integrated circuit.

* * * * *