PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : H04L	A2	(11) International Publication Number: **WO 98/26533**
		(43) International Publication Date: 18 June 1998 (18.06.98)

(21) International Application Number: PCT/US97/21536

(22) International Filing Date: 26 November 1997 (26.11.97)

(30) Priority Data:
60/031,897 27 November 1996 (27.11.96) US

(71) Applicant: 3COM CORPORATION [US/US]; 5400 Bayfront Plaza, Santa Clara, CA 95052 (US).

(72) Inventors: SHERER, William, Paul; 1054 Gardenia Way, Sunnyvale, CA 94086 (US). HART, John; 15121 Pepper Lane, Saratoga, CA 95070 (US). WANG, Peter, Si-Sheng; 10039 Minaker Court, Cupertino, CA 95014 (US). JAIN, Vipin, Kumar; Apartment 3201, 131 Saratoga Avenue, Santa Clara, CA 95051 (US).

(74) Agents: LeBLANC, Stephen, J. et al.; Townsend and Townsend and Crew LLP, 8th floor, Two Embarcadero Center, San Francisco, CA 94111 (US).

(81) Designated States: AU, CA, GB, JP, European patent (AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD AND APPARATUS FOR INTERMEDIATE SYSTEM BASED FILTERING ON A LOCAL AREA NETWORK UNDER END SYSTEM CONTROL

(57) Abstract

An intermediate system, end system and method capable of reducing unwanted packet traffic in a LAN is disclosed. The ES drivers send to the IS indications of whether or not they wish to receive a particular packet stream and the IS includes this information in a filter table. The IS then does not transmit packets to said ES when said ES has indicated it does not wish to receive said packets.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHOD AND APPARATUS FOR INTERMEDIATE SYSTEM BASED FILTERING ON A LOCAL AREA NETWORK UNDER END SYSTEM CONTROL

BACKGROUND OF THE INVENTION

This application claims priority from provisional patent application 60/031,897, filed on 11/27/96, which is incorporated herein by reference.

The invention relates to transmission of information between digital devices on a network. More particularly, this invention relates to a method and apparatus for allowing an end system node in a local area network to control the flow of packets directed towards the end system by directing an intermediate system at the data link/media access control layer to filter packets such that packets are only delivered to end systems that actually wish to receive them.

Technology related to aspects of the present invention and in part conceived by inventors of the present invention has been under collaborative development within the 802.1 working group of the I.E.E.E. beginning in approximately August 1995 for the purposes of developing a standard. This application further describes additional technologies that are related to, and improvements upon, the standard under discussion. No standard has been published.

Standards

This specification presumes working familiarity with the general concepts, protocols, and devices currently used in LAN networking applications and in WAN internetworking applications. A number of background concepts are discussed more fully in the provisional application cited above and other related applications cited herein. This discussion presumes familiarity with the protocols mentioned in these related applications. A set of protocols used for networking within a LAN referred to as the IEEE 802 protocol suite, available from the IEEE (Institute for Electrical and Electronics Engineers). These IEEE 802 protocols have been revised and reissued by the ISO (International Organization For Standardization) with the

designation ISO 8802. Among the protocols specified in IEEE 802 are IEEE 802.3, the LAN protocols commonly referred to as *Ethernet*. A separate set of protocols used in *internetworking*, i.e. connecting multiple LANs, may be referred to as the *TCP/IP Protocol Suite*. (TCP and IP are acronyms for Transmission Control Protocol and Internet Protocol.) The TCP/IP Suite is promulgated in a series of documents released by the Internet Engineering Task Force. The documents are referred to as RFC's (Requests For Comment) and are available via FTP at *ds.internic.net*. An brief overview of some concepts necessary for an understanding of the invention is presented below. For a more detailed discussion of background information, the reader should consult the above mentioned standards documents or a number of readily available reference works including Stevens, R. W., TCP/IP Illustrated, Addison Wesley, 1994.

Fig. 1

Fig. 1 illustrates a local area network (LAN) 40 of a type that might be used today in a moderate sized office or academic environment and as an example for discussion purposes of one type of network in which the present invention may be employed. LANs are arrangements of various hardware and software elements that operate together to allow a number of digital devices to exchange data within the LAN and also may include internet connections to external wide area networks (WANs) such as WANs 42 and 44.

Typical modern LANs such as 40 are comprised of one to many LAN intermediate systems (ISs) such as ISs 60-63 that are responsible for data transmission throughout the LAN and a number of end systems (ESs) such as ESs 50a-d, 51a-c, and 52a-g, that represent the end user equipment. The ESs may be familiar end-user data processing equipment such as personal computers, workstations, and printers and additionally may be digital devices such as digital telephones or real-time video displays. Different types of ESs can operate together on the same LAN. Note also that within a LAN such as 40, a WAN IS such as 64 or 66 might also be treated by the LAN devices as an ES, especially in the case where a WAN IS also serves as a

high-performance server for the network. In one type of LAN, LAN ISs 60-63 are referred to as *bridges* and WAN ISs 64 and 66 are referred to as *routers*, however many different LAN configurations are possible, and the invention is not limited in application to the network shown in Fig. 1.

The LAN shown in Fig. 1 has segments 70a-e, 71a-e, and 72a-e, and 73a. A segment is generally a single interconnected medium, such as a length of contiguous wire, optical fiber, or coaxial cable or a particular frequency band. A segment may connect just two devices, such as segment 70a, or a segment such as 72d may connect a number of devices using a carrier sense multiple access/collision detect (CSMA/CD) protocol or other multiple access protocol such as a token bus or token ring. A signal transmitted on a single segment, such as 72d, is simultaneously heard by all of the ESs and ISs connected to that segment.

In a LAN such as 40, data is generally transmitted between ESs as independent *packets* or *frames* as in known in the art. An ES such as 52g may transmit data with any other device on the LAN by transmitting a data packet containing a destination address for the intended destination. If the intended destination is directly connected to the same segment, such as ES 52d, then ES 52d hears and receives the packet as it is being transmitted by 52g. If, however, the destination ES is not directly connected to the same segment as the source ES, then LAN 40 is responsible for transmitting the data to a segment to which the destination ES is connected. Generally, a source ES is not aware of whether a destination ES in its LAN is directly connected to its segment. The source simply transmits the packet with a destination address and assumes that eventually the destination will hear the packet. Transmissions within the LAN are generally *source driven*, i.e. the LAN will deliver a data packet from a source to the destination address specified in the packet regardless of whether that destination ES actually wants to receive the packet. In general, packets contain user data that the user of an ES wishes to receive, such as portions of a data file or video or audio data stream which will be reassembled at the ES

after all packets that make that file are received or portions of a video stream which will be displayed to the user. In some prior art systems, the data packet may contain information that the ES formerly wishes to receive, but no longer wishes to receive, such as packets for a video conference that the ES is no longer connected to. Packets may also be *control packets*, containing control information that is used to facilitate communication within the network. In some protocols a control packet looks like the data packet shown in Fig. 3, with the data portion containing one or more protocol data units that are interpreted by the particular protocol.

Drivers, Adaptors, and LAN Topology

Each of the ISs and ESs in Fig. 1 includes one or more *adaptors* and a set of *drivers*. An adaptor generally includes circuitry and connectors for communication over a segment and translates data from the digital form used by the computer circuitry in the IS or ES into a form that may be transmitted over the segment, which may be electrical signals, optical signals, radio waves, etc. An ES such as 50b will have one adaptor for connecting to its single segment. A LAN IS such as 61 will have five adaptors, one for each segment to which it is connected. A driver is a set of instructions resident on a device that allows the device to accomplish various tasks as defined by different network protocols. Drivers are generally software programs stored on the ISs or ESs in a manner that allows the drivers to be modified without modifying the IS or ES hardware. LANs may vary in the *topology* of the interconnections among devices. Common topologies for LANs are bus, tree, ring, and star. LANs may also have a hybrid topology made up of a mixture of these. The overall LAN pictured in Fig. 1 has essentially a tree topology, but incorporating one segment, 72d, having a bus topology. A ring topology is not shown in Fig. 1, but it will be understood that the present invention may be used in conjunction with LANs having a ring topology.

Bridges

The LAN ISs in LAN 40 include *bridges* 60-63. Bridges are understood in the art to be a type of computer optimized for very fast data communication between two or more segments. For example, bridge 60 is a computer having a processor, a memory for storing network information, connections to two or more separate segments, and a buffer memory for storing packets received from one segment for transmission on another segment. Bridge 60 receives packets from a source segment such as 70f, stores the packets, and then transmits the packets on another segment such as 70a, when the bridge detects that the other segment is silent. A bridge according to the prior art generally makes no changes to the packets it receives on one segment before transmitting them on another segment. Bridges are not necessary for operation of a LAN and in fact in prior art systems they may be invisible to both the ESs to which they are connected and sometimes to other bridges and routers.

At its most simple, a bridge temporarily stores any packet data received on one of its connections, or *ports*, and then, as each other port is available, the bridge forwards, or *bridges*, the packet out of each other port. Even at this most simple level, a bridge such as 60 tends to isolate network traffic on segments and reduces the chances of collision between packets. Modern bridges, as described below, also provide *filtering* functions whereby a bridge learns the LAN addresses of all ESs that may be reached through each of its ports and forwards packets only out of the port to which the destination ES of that packet is connected. Filtering bridges are enabled to quickly examine the LAN address of every received packet to determine whether and to which segment that packet must be bridged. As an example, when filtering bridge 62 receives a packet on segment 72a addressed to 52b, that packet is bridged only to segment 72b and not to segments 72c and 72d.

In order to accomplish this *filtering* function, a bridge must somehow know which ESs are attached to each segment connected to the bridge. Generally, this is done in one of two ways: a bridge may be configured by a human network manager to

know the LAN addresses of the ESs connected to each segment, or a bridge may be enabled to learn the LAN address of ESs connected to each segment as the bridge is receiving packets. Bridges enabled to learn which ESs are connected to each of their segments do so by examining the LAN source address of packets received on a particular port. A self-learning bridge generally stores the information it learns from examining the source address of packets in a portion of the bridge's memory, sometimes referred to as a Bridge Filtering Table (BFT). Once a bridge has placed entries in its BFT, upon receiving a packet, the bridge will examine the LAN destination address of the buffered packet and if, according to the BFT, the destination address is on the same segment from which the packet was received then the packet has presumably already been received by the destination ES and the bridge discards the buffered packet. If the destination ES is on a different segment from the originating ES then the bridge *bridges* the packet by transmitting it on the destination ES's segment. If the destination address is not present in the BFT, then the bridge must bridge the packet to all other segments to insure that the proper ES receives the packet. In this way, self-learning bridges gradually learn more and more about the ESs connected to them and gradually reduce unnecessary data flow through the LAN. In a prior art bridge, construction of the BFT and subsequent filtering of packets is accomplished transparently by the bridge without the need for the ESs to be aware of the bridge or to transmit any control packets to the bridge.

Some prior art bridges implement an algorithm known as the Spanning Tree Algorithm which allows them to ensure that a segment that is connected to more than one bridge only receives packets from one of them. This algorithm is described fully in IEEE standard 802.1D. In this case, the bridges may be aware of the existence of other bridges on the network and may communicate control packets with those other bridges.

Routers

ESs within LAN 40 can communicate with any other ES in LAN 40 either directly if the ESs are on the same physical segment or through a bridge. However, if an ES wishes to communicate with an ES or other service on a different LAN, that data must be transmitted over a WAN such as 42. Fig. 2 depicts WAN 42. WAN 42 is a network of networks, or an *internetwork*. (The largest and most well known internetwork is the world-wide Internet.) WANs are generally comprised of a number of larger computers that are optimized for WAN transmissions, herein referred to as *routers* 64 and 68a-e. A router is a generally larger computer than a bridge, but, like a bridge, it too has a processor, a memory for storing network information, and connections to two or more separate segments. Some routers, like router 64, provide WAN services to a LAN and in addition can forward WAN packets through the mesh network to facilitate WAN communication. Other routers are multi-user multipurpose computers or file-servers that include routing functions. Still other routers are computers exclusively reserved for handling WAN data traffic.

Communication of WAN packets over WAN 64 via the routers is very different from packet communication within LAN 40 and occurs under a different protocol having a different addressing scheme. Unlike prior art bridges, prior art routers communicate *control packets* with every ES to which they are attached as well as to other routers in the WAN. A router uses information it receives via control packets and possibly configuration information supplied by a human operator to build a representation for itself of the network, which the router stores in a *routing table*. A router examines the WAN destination address of every packet it receives and uses information stored in its routing table to make an individual routing determinations about a packet based on the packet's destination address, other information in the packet's header, and the router's knowledge about the dynamic state of the WAN. Unlike a bridge, a router may make two different routing determinations for different packets with the same destination address based on the dynamic state of the WAN. A router such

as 64 is generally unaware of the presence of any bridges within a LAN to which it is connected and sends all data into the LAN as though router 64 was directly connected to each ES within the LAN.

Typically, a WAN such as 42 will have a different addressing scheme and different packet structure than that used in the LAN. Every ES in LAN 40 that wishes to receive packets from WAN 42 must have assigned to it a separate WAN address. In TCP/IP, WAN addresses are 32 bits long and are generally written in a *dotted decimal* notation having values from 0.0.0.0 to 255.255.255.255. Router 64 learns the LAN address and the WAN address of every ES in LAN 40 and translates packets and addresses between LAN 40 and WAN 42.

Fig. 3 depicts a packet as it may be transmitted to or from router 64 on LAN segment 73a. The packet is essentially an Ethernet packet, having an Ethernet header 202 and a 48-bit Ethernet address (00:60:8C:19:AA) 204, and an Ethernet trailer 230. Within the Ethernet packet 200 is contained, or *encapsulated*, an IP packet, represented by IP header 212, containing a 32 bit IP address 214 (199.35.126.34). Packet 200 contains a data payload 220 which holds the data the user is interested in receiving or holds a control message used for configuring the network.

Layers

A final background concept important to understanding the present invention is the concept of *layered* network protocols. Modern communication standards, such as the TCP/IP Suite and the IEEE 802 standards, organize the tasks necessary for data communication into *layers*. At different layers, data is viewed and organized differently, different protocols are followed, and different physical devices handle the data traffic. Fig. 4 illustrates one example of a layered network standard having a number of layers, which we will refer to herein as: *the Physical Layer, the Data Link Layer, the Routing Layer, the Transport Layer,* and *the Application Layer*. These layers correspond roughly to the layers as defined within the TCP/IP Suite. (The 802 standard has a different organizational

structure for the layers and uses somewhat different names.)

At the Physical Layer, data is treated as an unformatted bit stream transmitted from one transmitter to one or more receivers over a single segment. At the Data Link Layer (DLL), data is treated as a series of independent packets (at this layer often called frames), each packet containing its own destination address and fields specifying packet length, priority, and codes for error checking. In a specific ethernet implementation, the DLL is further divided into a lower media access control (MAC) layer and a higher link layer control (LLC) layer. At the Routing Layer (sometimes referred to as Layer 3), data is treated as a series of independent *routing packets*. At the transport layer, data is seen as a connection between two hosts on the network. Transport layer protocol in TCP/IP includes TCP and UDP. The Application layer includes programs that a user interacts with to use network functions, such as e-mail, ftp, remote login, or http. Data at the application layer is often viewed as files.

An important ideal in layered standards is the ideal of *layer independence*. A layered protocol suite specifies standard interfaces between layers such that, in theory, a device and protocol operating at one layer can co-exist with any number of different protocols operating at higher or lower layers, so long as the standard interfaces between layers are followed.

For purposes of clarity, the present discussion refers to network devices and concepts in terms of specific examples, namely *Ethernet* and *TCP/IP*. However, the method and apparatus of the present invention may operate with a wide variety of types of network devices including networks dramatically different from the specific examples illustrated in Fig. 1 and described below.

LAN Broadcast and Group Address Packets and WAN Multicasting

A packet in LAN may contain a destination address indicating delivery to just one node, referred to as *unicast*, or a destination address indicating a group of destinations, referred to as *multicast*. Multicast addresses are special

destination addresses reserved by the LAN protocol for packets sent to multiple destinations.

At layer 3, *multicast packets* are delivered only to those routers and end systems that request receipt of them. When running according to the TCP/IP Suite, routers and ESs may accomplish multicasting through a special protocol referred to as the *Internet Group Management Protocol* (IGMP). According to IGMP, a router such as 64 periodically queries nodes connected to it to report back to the router if they wish to receive any WMP streams. An ES that wants to receive a WMP stream will respond to this *IGMP Query* by sending an *IGMP Report* back to router 64. An IGMP Report lists a WMP address that the ES wishes to receive. An ES sends a separate report for each WMP stream it wishes to receive. In the art, it is sometimes said that the ES joins a multicast group each time it indicates to the router that it wishes to receive a particular WMP stream.

In many prior art LANs, layer 2 devices are not able to determine which ESs wish to receive multicast packets and so once a multicast enters a bridge, that packet is forwarded out of the bridge to every other segment. In the case of a heavy data stream such as a video link, this can result in a huge amount of unwanted LAN traffic.

One prior art solution to this problem would be to reconstruct LAN 40 and replace each of the bridges 60-63 with devices that function more as routers and participate fully in the overall IGMP protocol. This is an expensive proposition, however, increasing the cost of the LAN hardware infrastructure, LAN management, and likely decreasing the overall speed of the LAN. Another proposed solution is to have layer 2 devices listen on layer 3 multicast control traffic and infer from that which ESs wish to receive layer 3 multicast packets as discussed in co-assigned U.S. patent applications 08/542,157 (9764-70) and 08/761,574 (9764-70-1) incorporated herein by reference. This approach handles only layer 3 multicast packets, however, and has other complexities as described in the above referenced applications.

From the preceding it will be seen that what is needed is a LAN and network devices capable of preventing the

transmission through the LAN of unwanted packets.

SUMMARY OF THE INVENTION

The invention is an improved method and apparatus for transmitting data in a LAN. According to the present invention, an improved LAN is capable of filtering unwanted LAN traffic. According to one embodiment of the invention, a LAN bridging device and ES include driver components to allow a simplified communication of flow control packets between them when a device either wishes or does not wish to receive a particular packet stream. LAN bridging devices also send these flow control packets to other bridging devices on a proxy basis when one of their ESs requests a packet stream.

The invention may be combined with techniques for sniffing and filtering Routing Layer packets at the Data Link Layer as described in the above referenced applications.

A bridge, according to one embodiment of the present invention, accomplishes these advantages by including a bridge segment table or filtering database for storing filtering information.

The invention in a further aspect comprises a simplified state machine and protocol for communicating layer 2 control data between nodes. This protocol reduces the amount of state that must be kept about each group, eliminates the need to keep state about groups in which a participant is not interested and is valid in the face of any single message loss.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a diagram of a local area network of one type in which the invention may be effectively employed;

FIG. 2 is a diagram of a wide area network of one type in which the invention may be effectively employed;

FIG. 3 is a diagram of an example data unit comprising an IP packet encapsulated in an Ethernet frame that may be used to carry flow control messages in accordance with the present invention;

FIG. 4 is a diagram of an example layered network

protocol in which the present invention may be effectively employed;

FIG. 5 is a block diagram of an improved bridge according to the invention.

FIG. 6 is a diagram illustrating the operation of an improved registration protocol in a bridge and an end system according to one aspect of the invention.

FIG. 7 is a block diagram of an end system including an improved registration protocol according to the invention.

FIG. 8 is a block diagram of a switch/bridge including an improved registration protocol according to the invention.

FIG. 9 is a diagram of a computer system with a fixed medium 717 which may be used to implement one embodiment of the invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The following detailed discussion will describe the function of an improved bridge and end system according to the invention and a simplified flow control protocol in terms of specific protocol suites. It will be understood to those of skill in the art that the invention has application in variations on these protocol suites and in other protocol suites.

Fig. 5 is a block diagram of a bridge 62 with improvements according to an embodiment of the invention. The bridge has five ports 80a-e which provide circuitry and connections that enable the bridge to communicate segments 72a-e. Packets received over any port are temporarily stored in a memory such as Packet Buffer 82. Controller 84 reads each received packet and processes that packet based on the instructions specified in driver 86. Controller 84 includes connections (not shown) to each other bridge component for sending and receiving control signals.

Controller 84 maintains a Bridge Segment Table (BST) 85 (which may also be referred to as a Bridge Filter Table or as a filter database) which contains entries used by the bridge in performing its filtering function. BST 85 includes an entry

85a for each ES source address from which a packet is received and an indication of the port on which the packet is received. According to a default operation mode, a prior art bridge gradually learns about ESs to which it is connected by reading their source addresses from packets received on its ports. If a bridge receives a packet to a destination address for which there is not an entry 85a, that packet must be forwarded out of all ports. However, once a bridge has identified a particular ES address as connected to a particular port, it uses an entry 85a to look up that port and then only transmits the packet out of that port. However, prior art bridges generally do not include filtering mechanisms for group addresses or other attributes of packets and simply bridge group addresses out of every port.

According to the present invention, however, a bridge further is able to create attribute entries AT1 to ATn, such as 85b, that allow it to selectively filter packets other than unicast packets. In one embodiment, these entries are created in accordance with a layer 2 control protocol with an Applicant and a Registrar State Machine in each device for each attribute class (or group) as described below.

A bridge also may include entries such as 85c for IGMP multicast groups that are constructed in accordance with a method that infers multicast filtering by sniffing layer 3 multicast query and report packets as described in incorporated patent applications S/N 08/542,157 and 08/761,574 and an entry such as 85d that designates a port as a router port.

As shown in Fig. 5, entries in BST 85 are indexed according to an attribute. In a prior art learning bridge, this attribute may be understood to be the unicast destination of the packet. In the above-incorporated patent applications, this attribute may be an indicator for a layer 3 multicast group. In the present invention, this attribute may be any of number of different criteria, including layer 2 or layer group identifiers for the packet, a priority of the packet, a flag indicate a type of the packet (such as video), or any other criteria that a bridge may quickly determine from a incoming packet.

It should be understood that Fig. 5 represents just one possible organization for a filtering database 85 and a bridge 62. Many other embodiments are possible, including embodiments in which a filtering control engine is present for each port and the filtering database is distributed. In another embodiment, different filtering tables 85 are maintained for different types of filtering functions, each having different organizations.

An Improved Protocol Allowing an ES to Modify a Filtering Table in a Layer 2 Intermediate Device

This section describes a particular protocol according to the invention for communicating control packets for improved layer 2 filtering. The protocol allows a receiver (an ES or an IS), at layer 2, to signal to a sender that the receiver does or does not wish to receive certain packets. Stated differently, the protocol allows an ES to join or leave a group, where joining a group is requesting delivery of packets having certain attributes and leaving a group is requesting packets having certain attributes not be delivered. An ES or IS that has joined a group may be referred to as a participant in the group.

One protocol for accomplishing layer 2 group registration has been under discussion within an I.E.E.E. working group under the name GARP (originally Group Address Registration Protocol, later Generic Attribute Registration Protocol). According to the present invention, a further embodiment of the protocol provides a simpler state machine that is easier to validate.

In this implementation, a connection to a network segment includes an Applicant State Machine (ASM) for each active group. IS connections and optionally ES connections also include a Registrar State Machine (RSM) for each active group. The ASM sends required messages. The RSM is passive and does not send messages but affects the behavior of the ASM.

In one embodiment, a device may erase state about a particular group once an Applicant leaves a group. In many situations, this results in significant savings at a device in

terms of processing and memory overhead.

Protocol Messages

According to the present invention, the following message types are sent by an ASM for a group and are heard by ASMs and RSMs. A message includes an identifier for the group to which it applies. Where more than one group is active at a device, messages for multiple groups may be aggregated into a single flow control packet.

JoinEmpty: Applicant wishes to join this group and Applicant's corresponding Registrar (if a corresponding Registrar exists) has not registered the existence of another member on this segment.

JoinIn: Applicant wishes to join this group and Applicant's corresponding Registrar has registered the existence of another member on this segment.

LeaveEmpty: Applicant was a member of this group, but is now leaving.

A protocol according to the invention may also include the following message, which is not necessarily group specific:

LeaveAll: Sent by an IS or server to alert Applicants that the IS is timing out all memberships and Applicants should reregister memberships in which they are interested.

A different protocol also includes an **Empty** message, which an Applicant sends to say that it is not a member of a group and does not believe there are any other members on this segment. This message is sent when a registrar sees a leave message and does not see any other join messages. According to the current invention, this message is not needed.

Applicant Responsibilities

According to the present invention, a group Applicant is responsible for making sure the following conditions are met:

(1) this participant is registered as a member by other Registrars;

(2) if there are any other participants on the segment, they are registered as members by this participant's Registrar;

(3) if this participant leaves the group, other participants communicate to satisfy condition #1 and #2.

According to this aspect of the invention, all of the above conditions are met in face of any single message loss by making good use of the distinction between **JoinEmpty** and **JoinIn** messages. Receiving a **JoinEmpty** message tells an Applicant for that group that #2 is satisfied, but alerts the Applicant that #1 is not satisfied. In response, this Applicant must transmit **JoinIn** messages to satisfy #1. On the other hand, when the Applicant transmits **JoinEmpty**, the Applicant knows #1 is satisfied, but #2 is not. So, in response, at least one other Applicant (if there is any) should speak up and transmit **JoinIn** to satisfy #2.

According to one embodiment, a join timer (**jnTimer**) runs for the ASMs at a node. When this timer expires, join/leave protocol data units (PDUs) are aggregated and transmitted in a one or more frames if any ASM demands it.

In one embodiment, every node in the network may have a different timer and the protocol is designed to operate properly in the face of multiple independent timers.

In a particular embodiment, the join timer is random (0-max), max defined according to a standard; and a leave timer, described below, is set to be greater than 3 * the maximum possible JoinTimer. The larger the value of leave timer, the more time it will take before this participant starts not transmitting (pruning) traffic after receiving a first leave indication.

The state machine is not dependent on timer values of the participants. Though, in one embodiment the leave timer has to be greater than 3 * maximum join time, it is not dependent upon the value of join timer of any protocol entity, rather is dependent upon the maximum value of join timer.

Applicant State Machine Description

According to a specific embodiment of the invention, an Applicant State Machine for a group at a node connection has the following states:

VX (Very Anxious): #2 may have been satisfied, but #1 has not been satisfied yet, and there is no reason to believe that it will be satisfied in the future.

AX (Anxious): If no messages were lost, #1 would have been satisfied and probably #2 has been satisfied. If #2 has not been satisfied yet, it will be.

IN (In Group): #1 must have been satisfied (in face of a single message loss). #2 should have been satisfied, but if not, it shall be satisfied (again in face of a single message loss).

WL (Will Leave): Applicant will transmit a Leave message on next transmit opportunity and will start leaving, unless the user requests to rejoin the group.

LA (Leaving Applicant): Applicant has already transmitted a Leave message. If no messages were lost, #3 would have been satisfied. Will transmit another Leave message on next transmit opportunity to make sure that #3 is satisfied, and leave the group.

OUT (Not In Group): In this state, the participate in not in the group. In some applications, an ASM for a group will be erased and an ASM will be created for this group if a request to join the group (reqJoin) is received from the protocol user. If a reqJoin is received in this state, or in the WL or LA state, an ASM for this group starts out in the VX state.

The table below is a depiction of various states in the Applicant state machines and the actions that will be taken when the messages indicated on the left are received (s means send; -- means don't care; -x- means invalid transitions or

messages that are not processed).

APPLICANT PASSIVE STATE MACHINE FOR A GROUP

	OUT	VX	AX	IN	WL	LA
jnTimer Expired	-x-	sJ[E,I];AX	sJ[E,I];IN	-x-	sLE; LA	sLE;OUT
rJoinIn	-x-	AX	IN	--	--	--
rJoinEmpty rLeaveEmpty rLeaveAll	-x-	--	VX	VX	OUT	OUT
ReqJoin	VX	-x-	-x-	-x-	VX	VX
ReqLeave	-x-	OUT	WL	WL	-x-	-x-

Joining a Group

According to an enhanced protocol in accordance with the invention, when a participant's user (i.e. a higher layer application) signals to join a group, an ASM for that group is revived or created and starts in a very anxious (**VX**) state. The Applicant therefore strives to satisfy #1 and #2. According to the invention, in the very anxious state, the Applicant must either: transmit two join messages (either **JoinIn** or **JoinEmpty**, depending on the state of the Registrar); transmit one join message (**JoinIn** or **JoinEmpty**) and receive one **JoinIn**; or receive two **JoinIn** messages to be comfortable. Note that if the Applicant receives two JoinIn messages before the join timer expires, it does not send any join messages itself. Though transmitting two **JoinEmpty** messages satisfies only #1, it makes this Applicant comfortable, because, if there are any other Applicants on this segment, the **JoinEmpty** will make them uncomfortable, and cause them to transmit **JoinIn** messages, satisfying #2.

Leaving a Group

When a participant's user signals to leave a group, the Applicant state machine transmits a **LeaveEmpty** PDU (when it gets an opportunity to transmit) and starts timing out its own Registrar. If no join messages are then heard on the segment before expiration of the leave timer, this Applicant transmits another **LeaveEmpty** message and moves out. Transmitting two leave messages guarantees that all Registrars will hear the

leave message, even in the face of a single message loss, and will time out. If there are any Applicants on the segment, a **LeaveEmpty** makes them uncomfortable (**VX**), and causes them to transmit or listen for join messages that satisfy #1 and #2. The first Applicant who gets an opportunity to transmit, transmits a **JoinEmpty** to satisfy #1. This **JoinEmpty** makes other Applicants uncomfortable and causes at least one of them to transmit **JoinIn** messages to satisfy #2. If a join message is heard by the leaving Applicant before transmitting the second leave message, the second leave message is suppressed and the Applicant moves out of the group.

The Registrar

The Registrar registers the membership of groups. It listens to the messages sent by Applicants in devices on the segment and registers the existence of at least one device on the segment that is in the group. In an IS, the Registrar may interact with the filtering entries in the filtering database to reflect the group memberships, so that packets can be filtered or forwarded out of the appropriate ports. In one embodiment, there is one registrar for each port on a IS. An ES may have a registrar or may not. The Registrar has the following states:

IN: Registrar has seen at least one Applicant for the group on the segment.

L (Leaving): Registrar has seen a Leave message and needs to see some Join messages to be comfortable that there is someone interested in this group. Registrar has also started the Leave Timer and will move to Empty state once the Leave Timer expires.

MT (Empty): Registrar believes there is no one interested in this group.

The Registrar listens to PDU exchanges taking place on a segment. Receiving a **JoinEmpty** or **JoinIn** causes it to move to **IN**, while a Leave message moves it from **IN** state to **L**, and finally to **MT**, based on the value of a timer.

A timer (lvTimer) for each group Registrar state machine is started when the Registrar moves from **IN** state to **L**

state. If the Registrar hears a join message while the timer is running, the timer is stopped and the Registrar moves to **IN**. Otherwise, when the timer expires, the Registrar removes the membership and moves to **MT**.

Though the Registrar does not transmit any PDUs itself, it affects the behavior of the corresponding Applicant state machine. If the Registrar is not **IN** (it is **L** or **MT** states) and the corresponding Applicant transmits a join message, it transmits a **JoinEmpty**. If the Registrar is **IN**, the corresponding Applicant sends a **JoinIn**. This is illustrated in the Applicant state table above. The leave timer is long enough so that after seeing a leave message, other Applicants (if there are any) get an opportunity to transmit join messages.

Registrar State Machine For a Group

	IN	L	MT
lvTimer	--	MT	--
rLeaveEmpty, rLeaveAll	L	--	--
rJoinIn, rJoinEmpty	--	IN	IN

Figs. 6, 7, and 8, provide alternative illustrations of the invention. Fig. 6 illustrates the operation of a bridge 62 and an ES 51a according to one embodiment of the invention. Bridge 62 includes a protocol entity 86, which is part of the driver software. Data link layer protocol controller (LLC) entities 83a and 83b control packet flow for their respective ports at the data link layer, while MAC entities 88a and 88b control data flow at the MAC (media access control) layer. In the illustration, a frame is shown received by 80a and the forwarded in accordance with a forwarding process to port 80b. The port interface for ES 51a likewise includes a MAC entity, LLC entity, and an ES protocol controller 96.

Fig. 7 is a representation of an ES in more detail, showing communication paths between an Applicant 100a, a Registrar 110a, a ES driver controller 96, and interface with a

higher layer protocol user 97, timers 120, and an LLC interface to a segment connection 93a.

Fig. 8 is a representation of a bridge in more detail, showing for a port communication paths between a Proxy Applicant 101a, a Registrar 111a, a LeaveAll process 102a, a bridge protocol controller 86, timers 122, filtering database 85, and an LLC interface to a segment connection 89a. Applicant 101a is labeled a Proxy Applicant because it sends leave and join messages to other ISs and ESs on behalf of other attached devices.

The following are some of the important characteristics of the general protocol and of the specific protocol implementation according to the invention:

1. In steady state, when an Applicant starts out, it will transmit two **JoinEmpty** or two **JoinIn** messages (depending on the state of the corresponding Registrar) to join a group.

2. When someone leaves a group and there are more Applicants on the segment, it may take 3-6 messages before every Applicant is comfortable again. The usual case is, this Applicant transmitting a **LeaveEmpty**, another Applicant responding with a **JoinEmpty**, followed by two **JoinIn** messages from other Applicants. The best case is when this Applicant transmits a **LeaveEmpty**, and there is just one more Applicant on the segment, who, after seeing this **LeaveEmpty** message, transmits two **JoinEmpty** messages. The worst case is, when this Applicant leaves and there are still more than one Applicants interested in the group. This leaving Applicant transmits two **LeaveEmpty** messages (only if the first **LeaveEmpty** message gets lost or if the Applicant timer is too short compared to the Applicant timer of other participants), one of the other Applicants, after seeing this **LeaveEmpty** message, transmits two **JoinEmpty** messages, followed by two **JoinIn** messages from yet another Applicant.

3. When the last Applicant leaves the group, it will transmit two **LeaveEmpty** messages to get out.

4. The simplicity of Applicant State Machine is achieved at the expense of a few extra messages that are

exchanged when another Applicant someone joins or leaves the group. This should not be an issue on segments other than large shared segments, where these messages can be suppressed if there are some members already, by maintaining passive and observer states. Such large shared segments will become rare in advanced networks, and even if there are such segments, the simpler state machine will reduce the processing and memory overheads significantly from the corresponding switch ports at the expense of a few extra messages.

5. If a participant is participating in multiple groups, overhead is reduced, because PDUs may be packed for different groups before transmission.

6. The state machine is robust against all known single-packet-loss conditions.

7. The described protocol is the same for communications from ES to IS, from IS to ES, and from IS to IS, thereby simplifying the implementation and allowing instructions from an ES to be easily propagated and aggregated through an ISs before being communicated to other ISs.

Other advantages of the invention according to specific implementations include: the dynamic registration and propagation of multicast addresses in a bridged/switched environment (a significant improvement in bandwidth utilization); the fact that the proposed protocol can be used in conjunction with any network layer multicast registration protocol (such as IGMP and others); the forwarding of multicast data frames at wire speed without any CPU intervention; and administrative control over propagation of multicast addresses.

The invention may be embodied in a set of executable computer program code which may be stored into a fixed computer medium such as a disk, diskette, volatile memory or nonvolatile memory, or any other medium for storing computer code. In such a case, when such instructions are loaded and executed in an appropriately configured network intermediate system and end system, the systems will perform as described herein. A representation of such a system 700 is shown in Fig. 9 containing CPU 707, optional input devices 709 and 711, disk drives 715 and optional monitor 705. Fixed media 717 may be

used to program such a system and could represent a disk-type optical or magnetic media or a memory.

The invention also may be embodied within the circuitry of an application specific integrated circuit (ASIC). In such a case, the invention may be embodied in a computer understandable descriptor language which may be used by an ASIC foundry to create an ASIC device that operates as herein described. In this embodiment, computer system 700 may be understood as a computer system for reading instructions from media 717 for the manufacturing of an ASIC.

The invention has now been explained with reference to specific embodiments. Other embodiments will be apparent to those of skill in the art. In particular, method steps have been grouped and labelled as being part of various sub-methods in order to increase clarity of the disclosure, however, these steps could be differently grouped without changing the essential operation of the invention. It is therefore not intended that this invention be limited, except as indicated by the appended claims.

WHAT IS CLAIMED IS:

1. A method for reducing forwarding of undesired packets by a LAN intermediate system (IS) comprising:

receiving a layer 2 flow control packet from a destination end system (ES) at an IS, said flow control packet specifying an attribute of packets that the ES does or does not wish to receive;

using said layer 2 flow control packet to update a filtering database on said IS; and

using said filtering database to determine forwarding of packets by said IS.

2. The method according to claim 1 wherein said IS is a layer 2 bridge enabled to learn port destinations for layer 2 unicast packets by examining the layer 2 source address of received packets and recording port destinations on which said packets are received.

3. The method according to claim 1 wherein said attribute comprises a destination multicast address of said packet.

4. The method according to claim 3 wherein said attribute comprises a source address of said packet.

5. The method according to claim 3 wherein said attribute comprises a priority level of said packet.

6. The method according to claim 1 wherein said IS forwards packets out of ports without modifying the contents or header of said packets.

7. The method according to claim 1 wherein said layer 2 flow control message is one of: a join_in message, a join_empty message, and a leave message.

8. A bridge for use in a local area network

comprising;

a plurality of ports capable of transmitting and receiving data on a plurality of network segments;

a memory for buffering data received on said ports or waiting to be transmitted on said ports;

a bridge controller capable of reading source and destination addresses of a data packet received on said ports and capable of receiving flow control packets from a plurality of end systems connected to said ports;

a filtering database in which said controller stores the source address of a packet received at a port along with an identifier of said port and in which said bridge controller stores an attribute identifier contained in a flow control packet along with an identifier of a port on which said flow control packet is received; and

a forwarding process that receives packets from one port and forwards those packets out of other ports based on information in said filtering database.

9. The device according to claim 8 wherein said device operates in accordance with an Ethernet transmission protocol and includes a media access controller and a link layer controller for each port.

10. The device according to claim 8 wherein said bridge segment table further comprises:

a plurality of packet attribute entries, each said entry comprising a packet attribute identifier and indicators for ports of said bridge to which packets having a particular attribute should be forwarded.

11. The device according to claim 8 further comprising, for each port and for each attribute registered at each port:

a registrar for holding state regarding whether said port connects with a participant that is receiving said attribute or said port may be leaving said attribute group or said attribute is empty;

a proxy applicant for generating flow control messages out of said port when other ports indicate they are interested in an attribute group;

a timer for controlling some of the state transitions in said registrar.

12. The apparatus according to claim 8 wherein said plurality of destination address entries includes entries comprising a group destination address and indicators for ports of said bridge, said indicators indicating whether packets received at said bridge having said group destination address should be forwarded on said ports.

13. The apparatus according to claim 11 further comprising a join timer and a leave timer.

14. A local area network comprising;

a plurality of end systems, each with a connection to a network segment wherein said end systems are capable of transmitting on said segment a layer 2 flow control packet, said flow control packet comprising an attribute indication for packets said end system wishes to receive data; and

a plurality of bridges, each bridge having connections to at least two network segments, at least one of said bridges capable of determining the addresses of end systems connected to a particular segment by reading a source address from packets received on that segment and capable of forwarding addressed packets received at one segment to only those segments having an end system matching that address and further capable of receiving a layer 2 flow control packet from an end system specifying attributes of packets wish said end system wishes to receive.

15. The local area network according to claim 14 wherein said bridges are further capable of detecting multicast query packets received on a port and designating that port a query port and wherein said bridges are capable of determining a frequency of received query packets on a query port and

setting a timeout interval to be several times said frequency and wherein said bridges will designate a query port a non-query port when a query packet is not received during a timeout interval.

16. An end system adaptor driver comprising:

an application interface for receiving from an application layer indications regarding which packet streams are of interest;

a controller for examining packets received on a segment and examining the contents of said packets;

a flow control applicant for generating flow control packets to be transmitted out of said port to indicate whether a packet stream is of interest; and

a transmission interface for generating complete packets for transmission on a segment.

17. The device according to claim 16 further comprising:

a registrar for receiving flow control packets from a port and storing state information based on said flow control packets.

18. A method allowing an ES connected to a segment to join a group thereby receiving packets having certain attributes by modifying a filtering database in a layer 2 intermediate system comprising:

creating an applicant process and applicant state machine for a group in an end system;

transmitting layer 2 flow control protocol data units from said ES on said segment comprising only join messages and leave messages; and

listening to protocol data units on said segment for said group and responding with appropriate layer 2 flow control protocol data units.

19. The method according to claim 18 wherein said layer 2 flow control data units comprise:

a join_empty message indicating applicant wishes to join this group and

a leave_empty message indicating applicant was a member of this group, but is now leaving.

20. The method according to claim 18 further comprising:

creating a registrar state machine for said group, said registrar state machine holding state for said group and affecting said transmitting.

21. The method according to claim 20 wherein said layer 2 flow control data units comprise:

a join_empty message indicating applicant wishes to join this group and applicant's corresponding registrar has not registered the existence of another member on this segment;

a leave_empty message indicating applicant was a member of this group, but is now leaving; and

a join_in message indicating applicant wishes to join this group and applicant's corresponding registrar has registered the existence of another member on this segment.

22. The method according to claim 19 wherein said ES may also receive a leaveall message to alerting applicants that the IS is timing out all memberships and applicants should reregister memberships in which they are interested.

23. The method according to claim 20 wherein an applicant is responsible for making sure the following conditions are met:

(1) this participant is registered as a member by other registrars;

(2) if there are any other participants on the segment, they are registered as members by this participant's registrar;

(3) if this participant leaves the group, other participants communicate to satisfy condition #1

and #2.

24. The method according to claim 23 wherein said applicant state machine has the following states:

VX (Very Anxious) indicating condition #2 may have been satisfied, but #1 has not been satisfied and there is no reason to believe that it will be satisfied in the future;

AX (Anxious) indicating that, if no messages were lost, condition #1 would have been satisfied and probably #2 has been satisfied but if #2 has not been satisfied, it will be;

IN (In Group) indicating that condition #1 must have been satisfied even if a single message was lost and condition #2 should have been satisfied, but if not, it shall be satisfied;

WL (Will Leave) indicating that applicant will transmit a leave_empty message on next transmit opportunity unless a request is received to rejoin the group.

LA (Leaving Applicant) indicating that applicant has already transmitted a leaveempty message and will transmit another leave_empty message on next transmit opportunity to make sure that #3 is satisfied and leave the group;

OUT (Not In Group) indicating that the participate in not in the group.

25. The method according to claim 24 wherein said applicant state machine sends messages and makes state transitions in accordance with the following table, with r indicating receive over a segment; Req indicating a request to the applicant state machine from a higher layer user entity; jnTimer indicating expiration of a join timer; s indicating send; -- indicating don't care; -x- indicating invalid transitions or messages that are not processed; J[E,I] indicating a join_in or join empty message, depending on state of registrar:

	OUT	VX	AX	IN	WL	LA
jnTimer	-x-	sJ[E,I];AX	sJ[E,I];IN	-x-	sLE; LA	sLE;OUT
rJoinIn	-x-	AX	IN	--	--	--

rJoinEmpty rLeaveEmpty rLeaveAll	-x-	--	VX	VX	OUT	OUT
ReqJoin	VX	-x-	-x-	-x-	VX	VX
ReqLeave	-x-	OUT	WL	WL	-x-	-x-

26. The method according to claim 24 wherein when a user requests to join a group, an applicant state machine for that group begins starts in a VX state in which the applicant must either:

transmit two join_in or joinempty messages;

transmit one join_in or joinempty message and receive one JoinIn; or

receive two join_in messages;

before moving to an IN state.

27. The method according to claim 24 wherein when a user requests to leave a group, an applicant state machine for that group transmits a leave_empty message and starts a leave timer and if no join messages are heard on the segment for the group before expiration of the leave timer, said applicant state machine transmits another leave_empty message and moves to the OUT state.

28. The method according to claim 23 wherein said registrar state machine has the following states:

IN indicating registrar has seen an applicant for the group on the segment;

L (Leaving) indicating registrar has seen a leave_empty message and needs to see some join messages to know that there is another applicant on this segment interested in this group;

MT (Empty) indicating registrar believes there is no one interested in this group.

29. The method according to claim 28 wherein said applicant state machine sends messages and makes state transitions in accordance with the following table, with r indicating receive over a segment; lvTimer indicating

expiration of a leave timer; -- indicating don't care:

	IN	L	MT
lvTimer	--	MT	--
rLeaveEmpty, rLeaveAll	L	--	--
rJoinIn, rJoinEmpty	--	IN	IN

30. A fixed computer-readable medium containing computer program code that when loaded into an appropriately configured network intermediate system and run will cause the computer to perform the method of claim 1.

31. A fixed computer-readable medium containing computer-interpretable instructions describing a circuit layout for an integrated circuit that, when constructed according to said descriptions and placed into an appropriately configured computer system, will cause the computer to perform the method of claim 1.

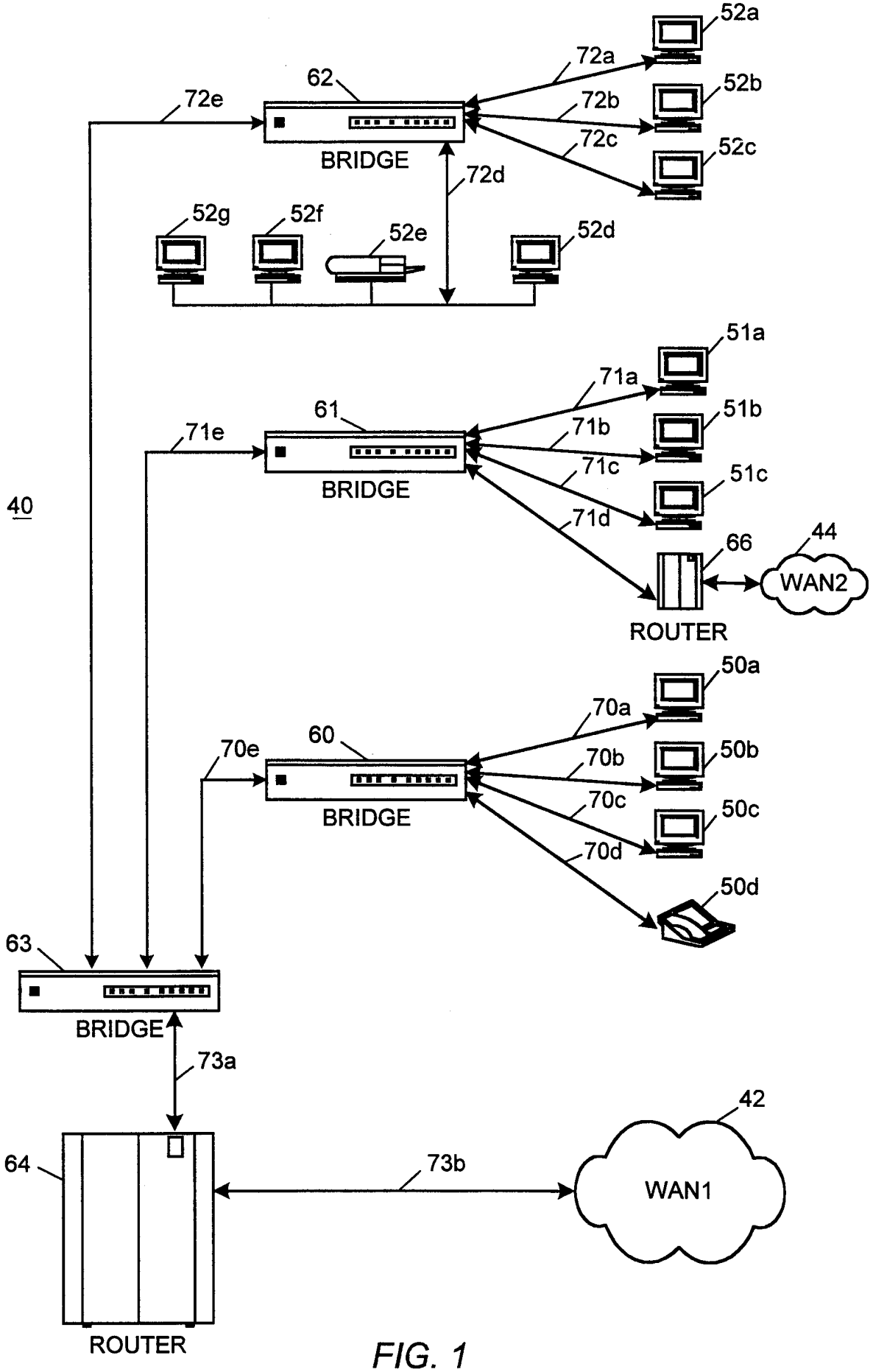
52a
72a
62
52b
72e
72b
72c
52c
BRIDGE
72d
52g
52f
52e
52d
51a
71a
61
51b
71e
71b
71c
51c
BRIDGE
40
71d
66
44
WAN2
ROUTER
50a
70a
70e
60
50b
70b
70c
50c
BRIDGE
70d
50d
63
BRIDGE
73a
42
64
73b
WAN1
ROUTER

FIG. 1

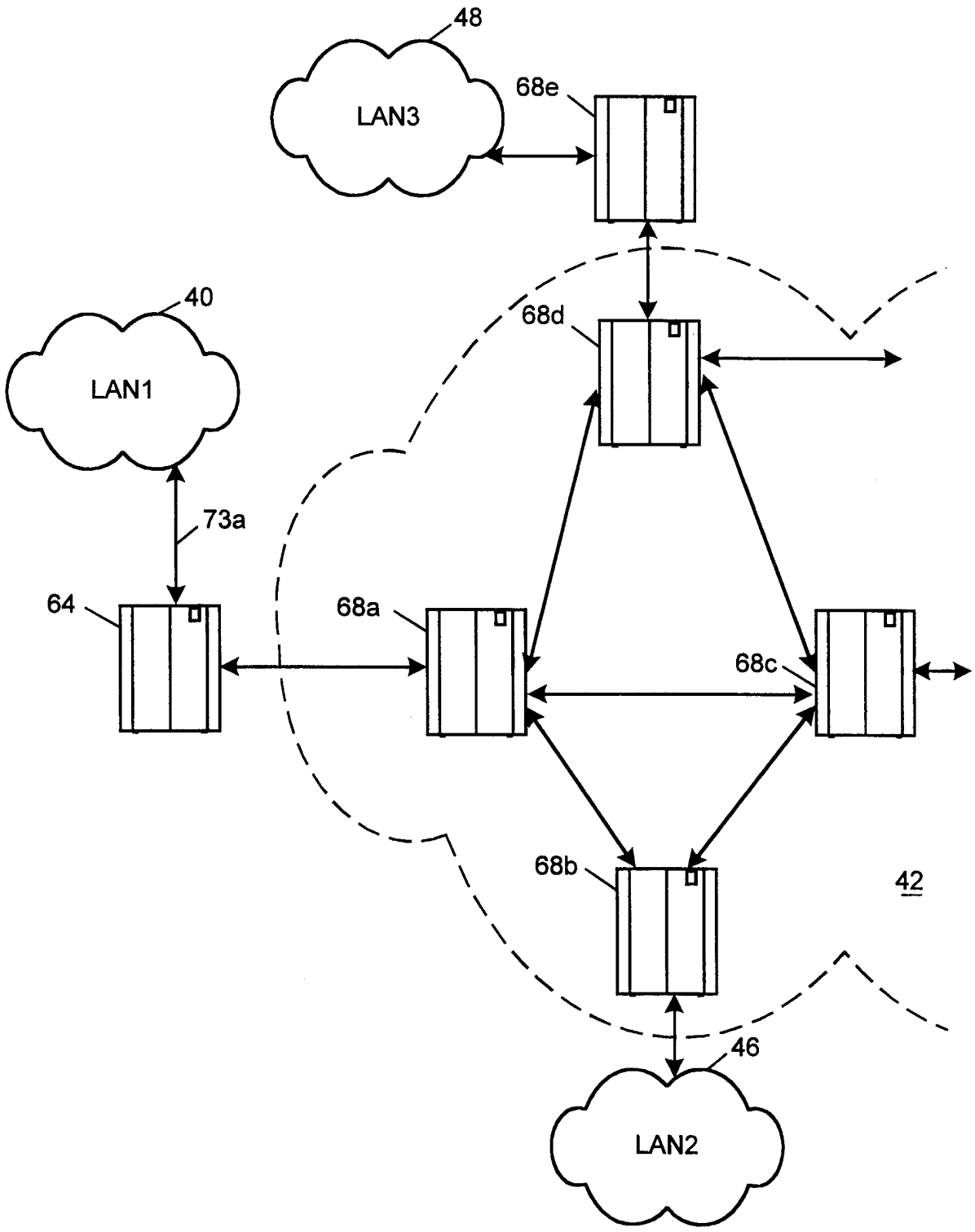

FIG. 2

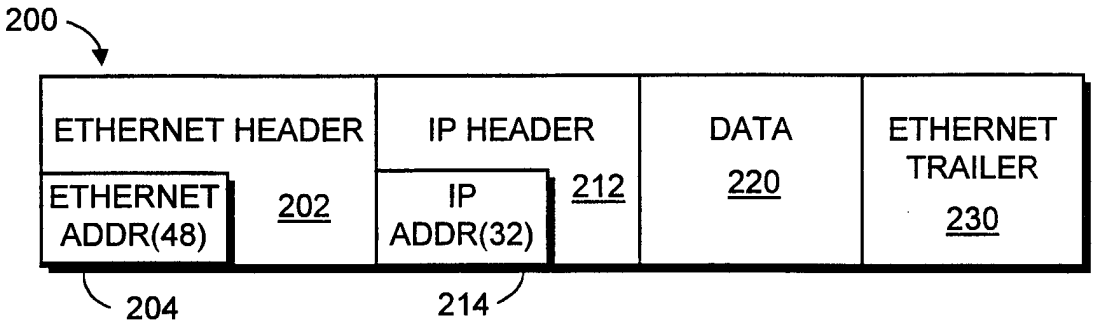

FIG. 3

HIGH

LAYER NAME (NUMBER)	DEVICES	DATA	PROTOCOLS
HIGHER LAYER PROTOCOLS			
APPLICATION LAYER (5)		FILES	FTP, HTTP
TRANSPORT LAYER (4)	ROUTERS	ROUTING PACKETS	TCP, UDP
ROUTING LAYER (3)	ROUTERS	ROUTING PACKETS	IP
DATA LINK LAYER (2)	BRIDGES	PACKETS/ FRAMES	ETHERNET
PHYSICAL LAYER (0,1)	REPEATERS	BITS	ETHERNET

LOW

FIG. 4

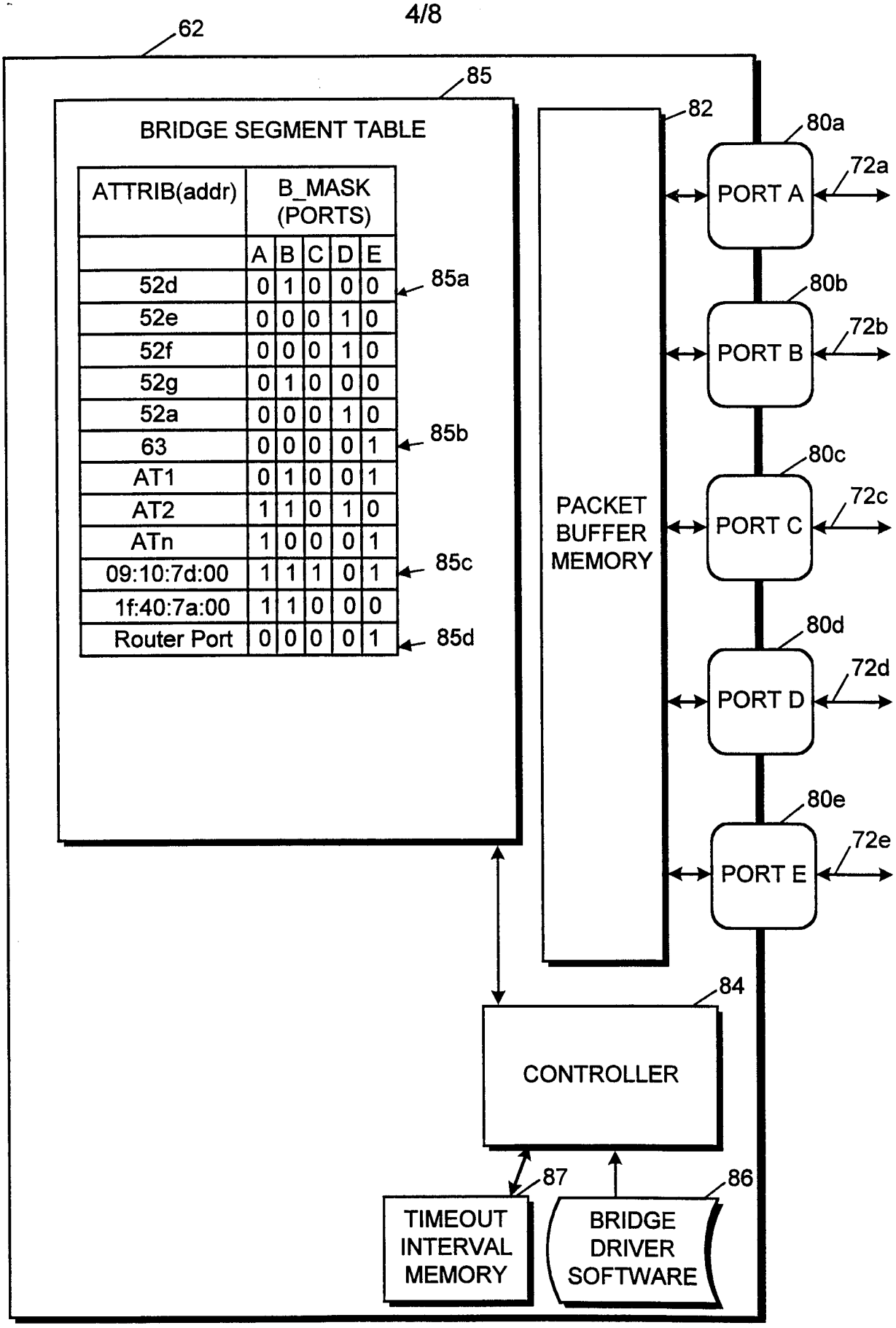
62
85
BRIDGE SEGMENT TABLE
ATTRIB(addr)
B_MASK (PORTS)
A B C D E
52d 0 1 0 0 0
52e 0 0 0 1 0
52f 0 0 0 1 0
52g 0 1 0 0 0
52a 0 0 0 1 0
63 0 0 0 0 1
AT1 0 1 0 0 1
AT2 1 1 0 1 0
ATn 1 0 0 0 1
09:10:7d:00 1 1 1 0 1
1f:40:7a:00 1 1 0 0 0
Router Port 0 0 0 0 1
85a
85b
85c
85d
82
PACKET BUFFER MEMORY
80a
PORT A
72a
80b
PORT B
72b
80c
PORT C
72c
80d
PORT D
72d
80e
PORT E
72e
84
CONTROLLER
87
TIMEOUT INTERVAL MEMORY
86
BRIDGE DRIVER SOFTWARE

FIG. 5

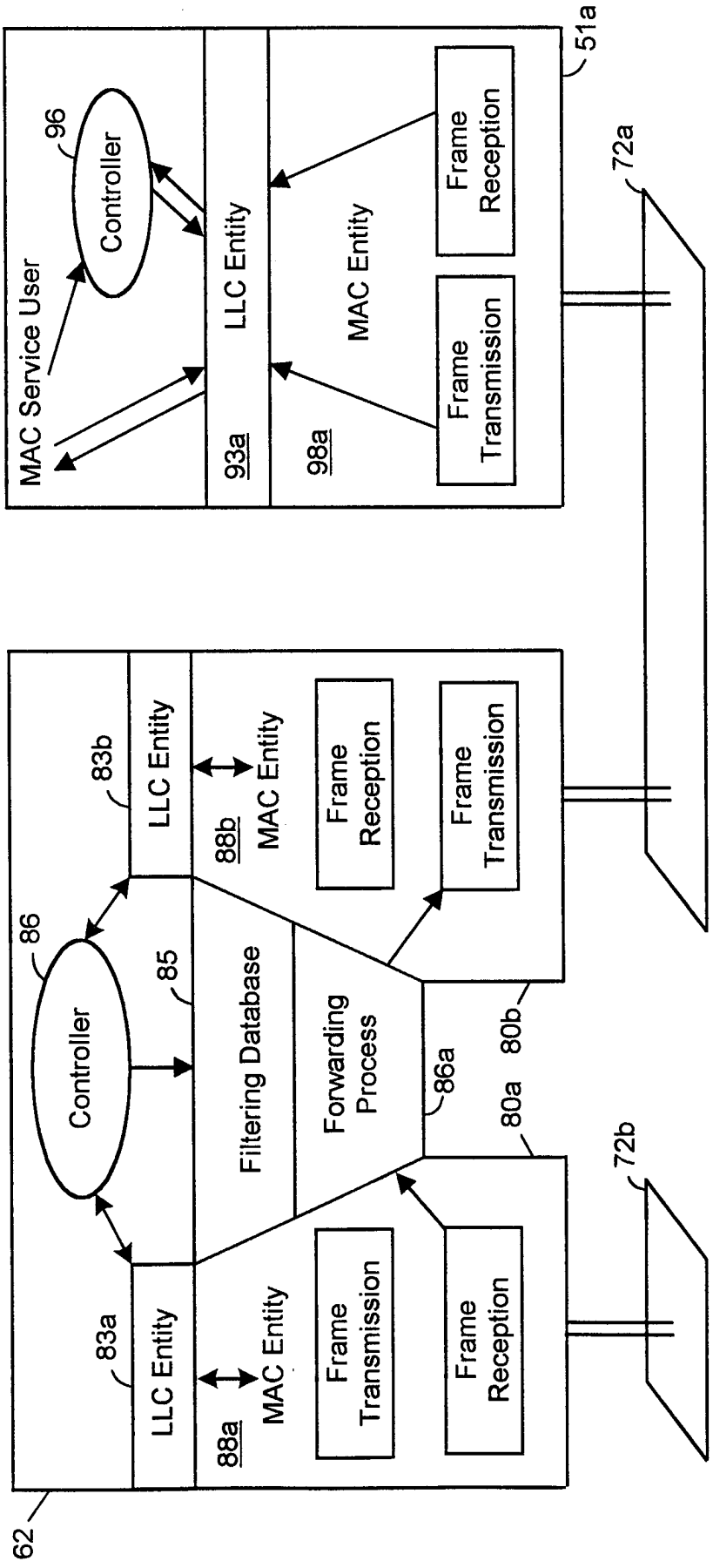
MAC Service User
96
Controller
LLC Entity
93a
MAC Entity
98a
Frame Reception
Frame Transmission
51a
72a
62
83a
LLC Entity
88a
MAC Entity
Frame Transmission
Frame Reception
83b
LLC Entity
88b
MAC Entity
Frame Reception
Frame Transmission
86
Controller
85
Filtering Database
Forwarding Process
86a
80a
80b
72b

FIG. 6

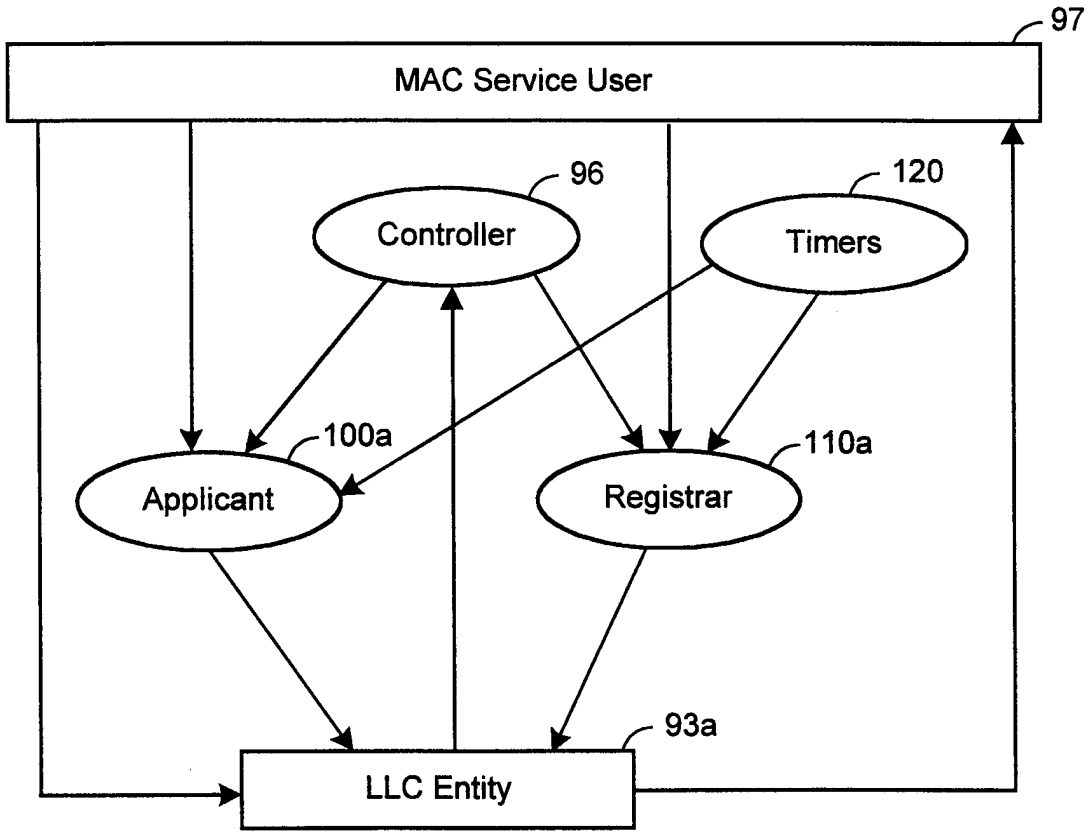
97
MAC Service User
96
Controller
120
Timers
100a
Applicant
110a
Registrar
93a
LLC Entity

FIG. 7

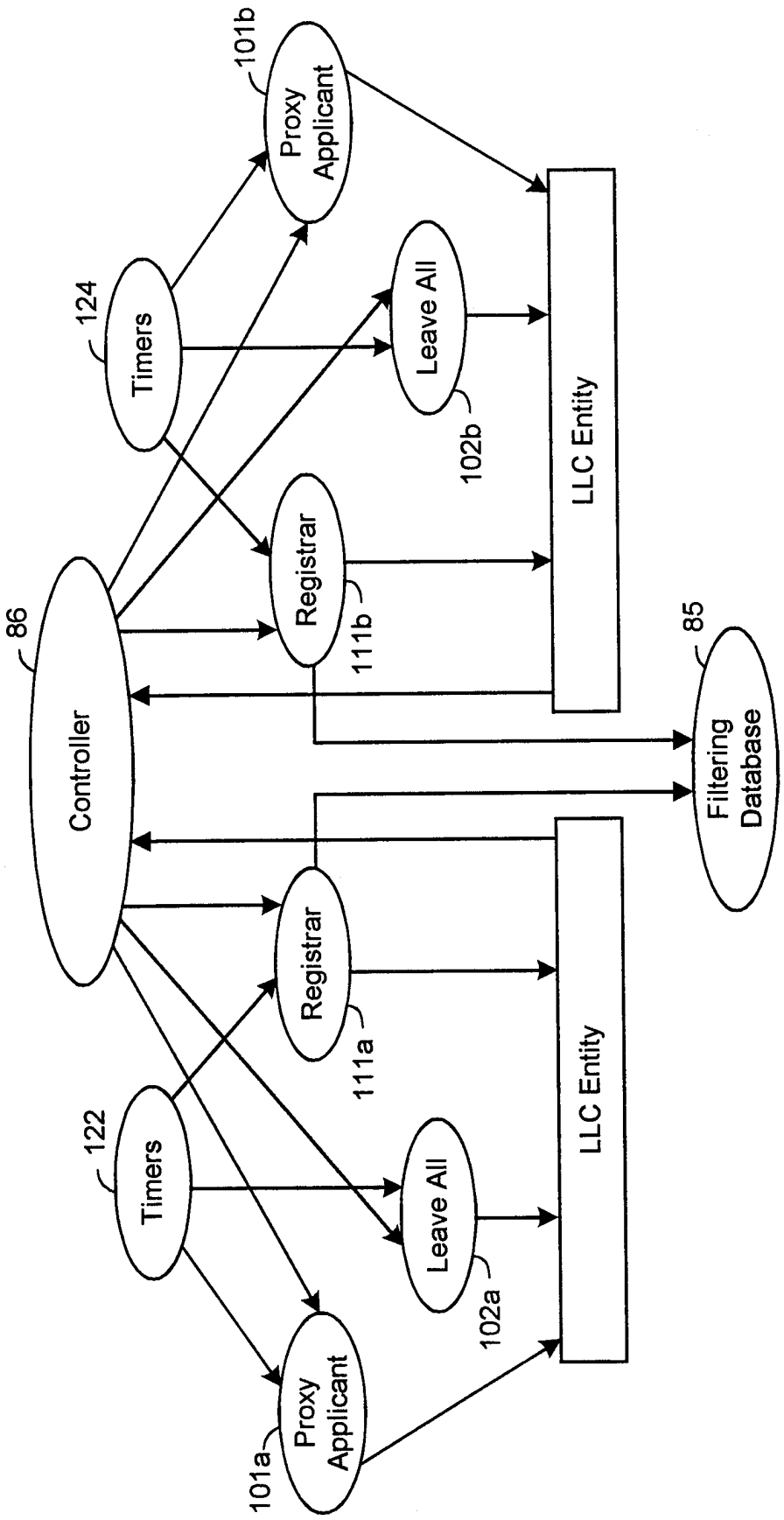
Controller
86
Proxy Applicant
101b
Timers
124
Leave All
102b
Registrar
111b
LLC Entity
Filtering Database
85
Registrar
111a
Timers
122
Leave All
102a
LLC Entity
Proxy Applicant
101a

FIG. 8

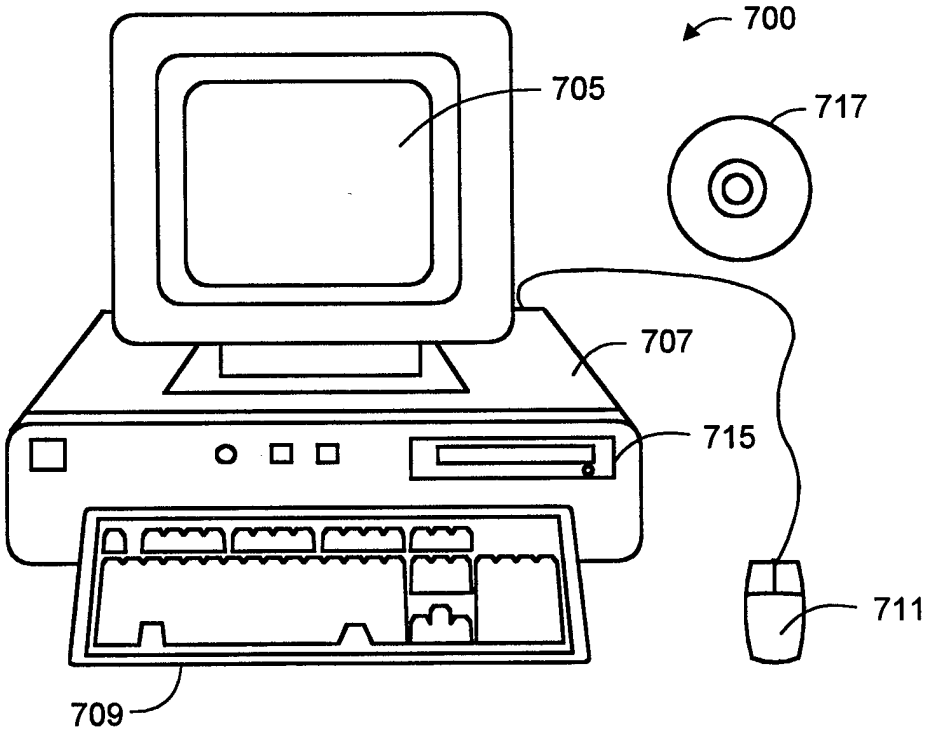

FIG. 9