Reprint

An Adaptable Frame Multiplexing Scheme for Source Routing Bridges

T. Antonakopoulos, J. Koutsonikos and V. Makios

Euromicro Journal of Microprocessing and Microprogramming

VOL. 35 NO. 1-5, SEPTEMBER 1992, pp. 409-416

Microprocessing and Microprogramming 35 (1992) 409-416
North-Holland

AN ADAPTABLE FRAME MULTIPLEXING SCHEME FOR SOURCE ROUTING BRIDGES

T. Antonakopoulos, J. Koutsonikos and V. Makios
Laboratory of Electromagnetics, University of Patras, 26500 Patras, Greece

An extension of the Source Routing algorithm is proposed for supporting multiplexed frame transmissions in the Extended LAN environment. The method exploits the new characteristics imposed by the introduction of high-speed backbone networks in the internetworking problem and allows better bandwidth allocation according to the current traffic load, without any modifications on the access protocol. The proposed enhancement uses the reserved bits of the Source Routing Control field and requires no modifications on the existing frame format. The multiplexing scheme is described in details along with a new statistic method for traffic monitoring. The proposed method can be easily implemented in hardware, so it can be included in a bridge implementation and a short description of such an implementation is given.

1. INTRODUCTION

The notion of the Extended LAN has been widely used to describe a collection of LANs connected by bridges [1]. Generally speaking, an Extended LAN contains various types of LANs, with incompatible Data Link formats and transmission characteristics. These LANs are interconnected using the so called bridge devices, which forward frames between the LANs transparently, implementing various address filtering algorithms. These filtering algorithms use the addressing information contained in the frame headers and their basic difference is based on the routing decision point [2].

In the Transparent Spanning Tree (ST) algorithm [3], each bridge keeps tables containing the station addresses in each attached LAN and forwards the received frames accordingly. This algorithm has been standardized by the IEEE 802.1 Committee and provides a stable, robust and simple to manage frame forwarding method. Another well-known technique is the Source Routing (SR) [4], originally proposed by the IEEE 802.5 Committee for multiple token ring interconnection. Following this method, the bridges are not required to keep address tables (for that reason, it is also called table-free bridging [5]) and the routing information is provided by the source station. A bridge uses the routing information field of each frame to decide whether to forward the frame or not. The bridge scans this field to test if its number is contained between the numbers of two LANs attached to it. In that case, the frame is forwarded, otherwise it is rejected. These two schemes have been extensively analyzed in the literature [6] and their characteristics have been highlighted.

In this work, we propose an enhanced version of the Source Routing method taking into account the new characteristics of the Extended LAN, when at least one high-speed network is attached. Especially, we consider the case where the high-speed network is used as a backbone to interconnect various LANs. Our work uses the reserved field of the original Source Routing method, imposes no frame overhead and allows the allocation of more bandwidth to bridges with increased traffic load without any modification to the access protocol. The basic idea is to exploit the differences between the largest allowed sizes on the traditional LANs and the high-speed network and to use multiplexed frames for inter-bridge transmissions. In the case where the high-speed access protocol is based on the number of frames allowed to be transmitted in each access, our method gives more bandwidth to the heavy loaded bridges. The term 'multiplexing' is used to denote the use of more than one received frames to formulate one output frame, while the term 'adaptable' is used to describe the bandwidth allocation procedure to the various bridges according to their current load.

In section 2 a brief description of the Source Routing method is given, while the multiplexing Source Routing extension is explained in section 3. In section 4, the mechanism implementing the Source Routing multiplexing inside the bridges is described in details. The 'Statistic Window' method used for traffic monitoring and internal buffer allocation is also described, giving emphasis to the implementation complexity implied by the method. Finally, a proposal for the hardware implementation of the multiplexing mechanism is given and the architecture of a bridge with such an implementation is also described.

2. THE SOURCE ROUTING ALGORITHM

The basic characteristic of the Source Routing algorithm is that each frame must contain a descriptor of its routing path from the source up to the destination station. The descriptor must contain the LANs and bridges numbers, the frame must pass through. The bridge makes the decision to forward a frame, based on the contents of the Medium Access Control (MAC) header of the frame. The transmitting station is responsible to specify the route and include the subnetwork (S) and bridge (B) numbering. The route descriptors consist of route designators [5], where each designator contains the LAN number and the next bridge number. Each route descriptor is the result of a route detection procedure started from the source station by broadcasting frames into the network to explore all possible routes to the destination station. When the destination station receives a route detection frame, it sends it back to inform the source station about the detected path. Once a route is known, the respective descriptor is appended to the transmitted frames in that route. In the Source Routing technique, the bridged network is loaded with Routing Detection frames, transmitted by the stations in a broadcast way, but this happens only once for each pair of stations. The main disadvantage of the Source Routing technique is its incompatibility to the ISO model, since the source station is mainly responsible for the route specification, but it is preferred in high traffic conditions because of bridge bottleneck avoidance.

The frame format in Source Routing contains the usual MAC fields and an extra field, the Routing Information (RI) field, which is of variable length (Fig. 1a). The presence of the RI field is indicated by the first bit of the Source Address field, which is the individual/group field. Since the Source Address is always an individual address, this bit indicates, when set, the presence of the RI field. As it is shown in Fig. 1b, the RI field contains a Routing Control (RC) field and a variable number of Route Designators (RD's). Each Route Designator consists of a *Subnetwork / Bridge* pair, thus all the RD's contain the route of the frame. The number of RD's in a frame is indicated by the L bits (Fig. 1c) of the Routing Control field and it is limited to 14.

The B subfield of the RC indicates an 'All-Routes Broadcast' frame, a 'Single-Route Broadcast' frame, or a 'Non-Broadcast' frame. The broadcast frames do not contain Route Designators, since they are mainly used for Routing Detection. They travel multiple paths to reach their destination and the RD's are updated by the bridges they pass through. The non-broadcast frames have a specific route to travel, indicated by an individual RD's sequence in the Routing Information field. The D bit of the RC

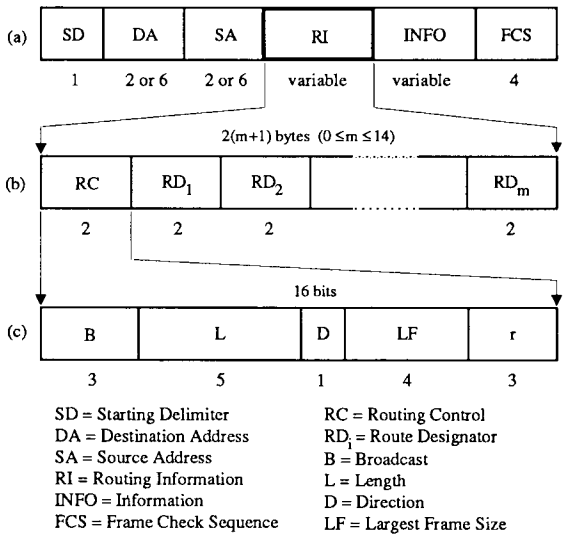

Fig. 1. The Source Routing Frame Format

field indicates the direction of the frame and the LF subfield indicates the maximum frame length permissible for a specific route. The Routing Control field contains also 3 reserved bits, not used for routing information.

3. THE MULTIPLEXING MECHANISM

Lately, the notion of the Extended LAN has changed, because new types of LANs are used in the structure of an Extended LAN and new transmission characteristics must be considered. Usually, a high-speed network is used to interconnect various LANs and the interconnection mechanism must utilize the new features. Our method is based on the fact that the frames transmitted using the Source Routing method have the minimum of the largest allowed size of frames that may be transmitted in a specific route, while the high-speed network can usually support longer frame lengths, a characteristic which is not exploited. On the other hand, the access protocols of the high-speed networks are usually based on the number of frames that can be transmitted in each access. The use of the number of frames per access, instead of the number of bytes, results to unfair and inefficient bandwidth allocation, especially during bursty traffic conditions and instantaneous peak traffic load in some bridges. For this reason we introduce the notion of the 'short path', which is the path between two bridges of a subnetwork. When bursty type of traffic is the main concern, the traffic characteristics imply the use of a multiplexing scheme during the peak traffic load.

Using the Source Routing technique for bridge implementation, it is possible to multiplex non-

broadcast frames, having a common 'short path' in the Routing Information field, in a longer frame, for the duration of the 'short path' transmission. More precisely, when a Bridge B_i recognizes the sequence $S_iB_iS_j$ in the Routing Information field of a frame received from subnetwork S_i, it will capture the frame to forward it to subnetwork S_j. This Bridge has also the ability to recognize the next Bridge, where the frame will be received, by examining the next Route Designator of the RI field. If, during a predefined time interval, the bridge B_i recognizes in a short that more than one frames have the same sequence $B_iS_jB_j$ in the RI field and their total length is lower than the maximum permissible length of the subnetwork S_j, it can multiplex them in a longer frame, since there is no reason for those frames to hold their independency in this 'short path' transmission, that is from bridge B_i to bridge B_j through the subnetwork S_j.

The multiplexing is valid for non-broadcast frames only, since the broadcast frames are Routing Detection frames and the sequence $B_iS_jB_j$ is not met inside the Routing Information field of a frame received in bridge B_i from subnetwork S_i. In addition, a broadcast frame could have a destination address for a station of subnetwork S_j, which cannot be detected by a Source Routing bridge.

The multiplexing of non-broadcast frames, having a common 'short path', is realized using the reserved bits of the Routing Control field. If those bits are all 0's, they indicate a non-multiplexed frame. The multiplexing bridge constructs a new frame, with a new Routing Information field, where the reserved bits indicate the number of multiplexed frames. Since the multiplexed frames are always more than one, a value of 1 at the reserved bits could indicate two multiplexed frames, thus, up to 8 frames could be multiplexed at any time.

Fig. 2 describes the format of the multiplexed frame, as well as the format of the Routing Information field and the format of the Routing Control subfield. The fields Destination Address (DA) and Source Address (SA) of the multiplexed frame may have any value, but the DA must be carefully selected, not to be the address of a subnetwork S_j station. For this reason, the DA and SA fields of the multiplexed frame 'borrow' their value from one of the individual frames being multiplexed. This guarantees also that the SA field has its individual/group bit set.

The Routing Information field of the multiplexed frame has a 'Non-Broadcast' indicator (the B subfield bits of the RC are all 0's) and it contains only the 2 Route Designators related to the 'short path', which is common to

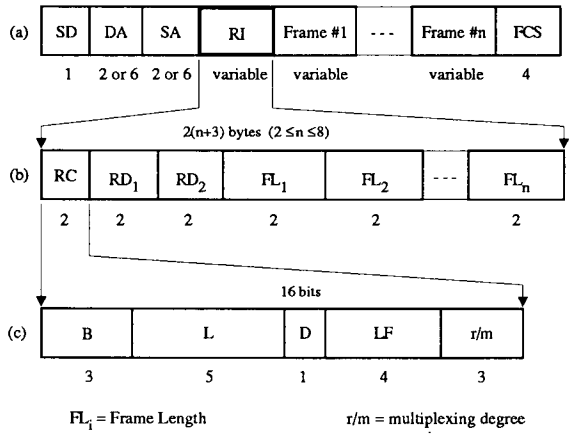

Fig. 2. The Multiplexing SR Frame Format

the individual frames. In addition, instead of more RD's, it contains the lengths of the individual frames (Frame Length fields), for easy demultiplexing in the receiving bridge.

The L bits of the Routing Control field of the multiplexed frame indicate the total length of the Routing Information field, comprising the RC field length (2 bytes), the 2 RD's for the 'short path' (4 bytes) and the individual frame lengths (2·r bytes, r being the number of multiplexed frames, indicated by the reserved bits).

Following the Routing Information field, the Information field of the multiplexed frame contains the individual frames, including their Destination and Source Address fields and their individual Routing Information fields, but without their FCS. This results to easier decomposition of the multiplexed frame to the initial individual frames. An FCS field is appended to the end of the multiplexed frame, to cover the entire frame.

The total overhead inserted in the subnetwork with the multiplexing depends on the degree of multiplexing, that is the number of individual frames contained in a multiplexed frame. If the degree is r, the overhead can be computed as:

Overhead = (DA) + (SA) + (RC) + (2 RD's)
+ 2·r + (FCS) - r·(FCS),

where the parentheses indicate the length of their contents.

For typical values of (DA) = (SA) = (RD) = 2 bytes and (FCS) = 4 bytes, the overhead is only 14 - 2·r bytes. This overhead is negligible and has no effect on the frame transmission time.

When a Source Routing bridge receives a frame, it checks the reserved bits of the Routing Control field, to recognize if the frame is a multiplexed one. In the case of a multiplexed frame, the bridge examines the two Route Design ators of the Routing Information field, to decide whether to forward the multiplexed frame to the next

subnetwork or not. If it recognizes itself as the receiving bridge, it starts the demultiplexing operation.

The number of multiplexed frames is the one indicated by the reserved bits plus 1 and the lengths of the individual frames are contained in the Routing Information field, following the second Route Designator. Using this information, the receiving bridge buffers the individual frames and then discards the Destination and Source Address fields of the multiplexed frame, the Routing Information field indicated by the L bits of the Routing Control field and the FCS field of the multiplexed frame, which is at the end of the received data. Then the receiving bridge examines the individual frames for further multiplexing, according to their next 'short paths' and then forwards them to the next subnetwork.

4. THE PROPOSED MECHANISM

In this section we study the internal organization of a bridge and we propose an architecture which can be used in bridges implementing the Source Routing multiplexing method. The existed LAN interconnection devices are generally considered to have the architecture shown in Fig. 3a. For each data flow direction, there is a Receive Queue (RQ), an Internal Queue (IQ) and a Transmit Queue (TQ). The RQ buffer stores any frames received from the attached subnetwork, while the IQ buffer contains only the frames to be forwarded. The TQ buffer contains the frames which are available for transmission to the next subnetwork. The MAC processor receives the frames in 'promiscuous' mode and stores them into the RQ. The bridge processor scans the received frames and determines their route using the Route Designators. If the frames have to be forwarded, the processor stores them logically in the IQ buffer and, depending on the next network interface availability, they are transfered logically to the TQ buffer. By 'logical store' we mean the update procedure of the internal buffer descriptors and not the actual frame transfer, considering that the MAC processors and the bridge processor share the same memory. In this discussion, the two special purpose processors implementing the two MAC interfaces are used to receive the frames from the network or to transmit them when they take the access control.

The traffic between the various interconnected LANs consists of frames generated in a bursty manner and of interactive type frames. The frames generated by bursty type traffic have the same destination for the duration of a burst and their multiplexing during the transmission in the network can improve the network bandwidth allocation according to the user needs. For a low complexity implementation of the multiplexing scheme, the architecture shown in Fig. 3b is proposed. This architecture uses a dynamically adaptable buffering scheme, which takes into account the traffic characteristics and reorganizes the allocated buffers accordingly.

This architecture uses more than one logical Internal Queues for the LAN to High Speed Network direction. Each IQ is devoted to a specific 'destination' bridge. When a frame arrives from the source LAN, it is stored into the RQ until the bridge processor decides to forward it or reject it. In the first case, the destination bridge is recognized and the frame is stored into the respective IQ. The IQ stored frames are then multiplexed to formulate a new frame, since the allowed frame length of the High Speed Network is, in most cases, greater than the allowed frame length of the interconnected LANs. The IQ frames are transfered to the TQ buffer after the completion of the multiplexing procedure.

In the destination bridge, each multiplexed frame is demultiplexed when it is stored in the IQ buffer. It must be emphasized that a frame

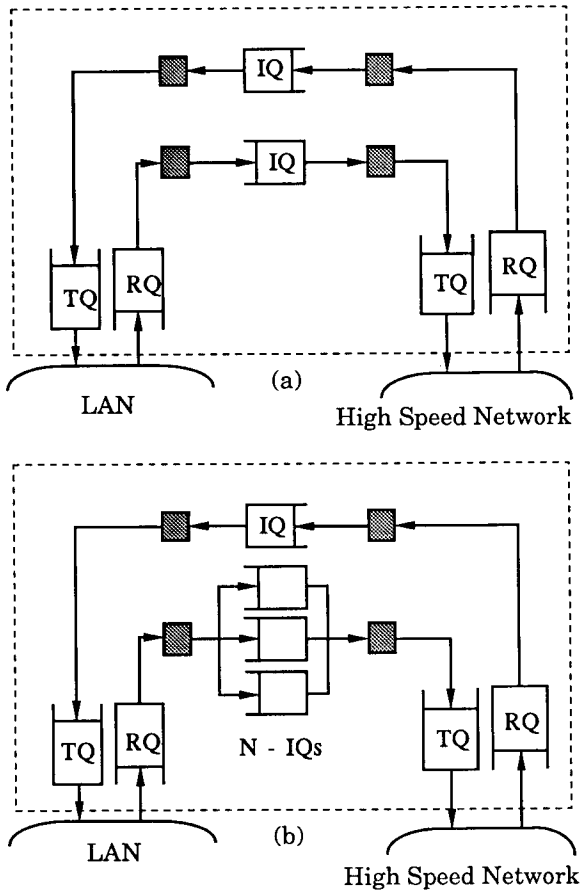

Fig. 3. Bridge Architectures : (a) single IQ
(b) multiple IQs

can be rejected, due to buffer overflow, only when it is received from the network. In the internal transfers there is no possibility of rejection, since the transfers are logical, the received data remain in the same physical area and only the respective buffer descriptors are updated. In the 'initial memory allocation' procedure, the bridge operating system reserves the memory which is required for the received frame length plus the memory required for the respective protocol headers. Each frame is associated with one or more 'buffer descriptors', which contain its length, the starting address and status bits. The TQ buffer of a source bridge handles each t-frame using the 'buffer descriptors' of the received frames.

The Number of Buffers

From the implementation point of view, the number of available IQ buffers is a crucial parameter and, as this number increases, the complexity of the buffer handling also increases. From experience in existing installations, the number of connected bridges in High Speed Network configuration is considered to be 'large enough' and increases rapidly as the speed and the covered area of the High Speed Network increase. Consequently, the number of the IQ buffers must be first considered.

The initial approach to the problem, regarding the number of IQ buffers, is to use the same number (minus one) as the number of bridges, in order to have a dedicated IQ buffer for each possible destination bridge. For this purpose, the bridge uses a 'learning capability', to allocate a logical IQ buffer for each detected 'short path'. The disadvantage of this method is that it becomes very complicated as the number of the connected bridges increases and the required processing time degrades the throughput of the bridge.

In order to overcome this disadvantage and approach a more realistic model, the following architecture has been developed. There are N+1 predefined logical IQs. The N IQs are buffers of variable length, allocated to the most active 'short paths', while the (N+1) IQ acts as a 'bypass'. The bridge uses its 'learning capability' to update the internal counters and uses a 'Statistic Window' method to route the received frame to the appropriate IQ. Each of the N IQs is allocated to a specific bridge temporarily and the allocation changes under the supervision of the 'Statistic Window' Handler. If the destination address of a frame does not match with any of the currently used destinations of the N dynamically allocated IQs, the frame is stored in the (N+1) IQ for further processing in which no frame multiplexing is performed. The frame multiplexing serves the need of implementation simplicity and the need of utilizing the available bandwidth efficiently. The proposed method combines the capability of dynamic buffer allocation with the deterministic organization of the buffer and requires a small percentage of the available processing power in its software implementation. Furthermore, the small complexity of the multiplexing algorithm makes feasible the hardware implementation of this method.

The 'Statistic Window' Method

As previously described, there is a predefined number of IQ buffers inside the bridge to handle the transfered information. In order to decide in which bridge an IQ buffer must be allocated, the 'Statistic Window' method is used [7]. From the included 'learning capability', each time a new address is recognized, a Destination Counter for that address is generated and is inserted into the structure of the Destination Counters. These Destination Counters are used to determine the allocation of each IQ buffer to a specific bridge. These counters indicate the number of address detections during a 'time window'. This 'time window' covers the last M processed frames of the RQ buffer.

Suppose now that there are T recognized bridges and the system has N available IQs for dynamic allocation. The counters form a 'window vector', which indicates the value of each counter when a received frame is processed. When the service of a new frame is completed, the 'time window' shifts one position and a new 'window vector' is generated. This 'time window' shift expresses the variation to the state of the counters at the frame arrival time. The counter associated with the new frame increases, while the counter associated to the last frame of the 'Statistic Window' decreases.

As previously mentioned, the operation of the 'Statistic Window' and the allocation of the IQs is performed by the 'Statistic Window' Handler. This Handler can be a software submodule in the bridge organization or a hardware implementation, which is activated by the bridge processor, when the processing of a frame is under completion. Suppose that counter k was decremented by 1, counter j was incremented by 1, as shown in Fig. 4, and that $n_{i,min}$ is the lowest counter value allocated to an IQ buffer at time i. After the Destination Counters' update, the buffer reallocation procedure begins. The Destination Counters are sorted in increasing order and the value of the $n_{i+1,min}$ is calculated. If two or more counters have the same value, the sorting is based on the time when this particular value was reached and the type of variation. The counter, which first received this value, has the lowest position. If two counters receive the same value at the same time, the counter who had the lower value in the preceding instant will have the

higher position in the new sorting list.

After the sorting task, the 'Statistic Window' Handler decides if an IQ buffer must be deallocated from a currently associated bridge and allocated to a new one. The 'Statistic Window' Handler is used by the bridge processor to determine the current frame's IQ destination buffer and probably to decide if the buffer reallocation requires a frame multiplexing completion. If no buffer has been deallocated after this procedure, the system remains in the previous allocation state and the normal frame processing continues.

When the 'Statistic Window' Handler starts the multiplexing process with the first two frames, it creates a new header for the multiplexed frame, comprising the Source and the Destination Address of the first multiplexed frame and a new Routing Information field. In this Routing Information field, the L bits of the Routing Control subfield have the value of 10 (two bytes for the RC subfield, four bytes for the 'short path' RDs and four bytes for the lengths of the two first multiplexed frames), the reserved bits of the RC have the value 1, indicating 2 multiplexed frames, the first two RDs contain the 'short parth' to be followed and the Frame Length (FL) indicators contain the lengths of the two first multiplexed frames. Each time a new individual frame is merged to the multiplexed frame, the value contained in the L bits of the new Routing Information field is incremented by 2, the value contained in the reserved bits is incremented by 1 and the length of the new individual frame is appended to the Routing Information field. It is obvious that following this method, the bridge allocates its buffering resources according to the incoming traffic characteristics.

During network operation, a frame multiplexing procedure is completed when one of the following conditions is met:

i) the maximum frame length of the High Speed Network has been achieved,

ii) the Multiplexing Waiting Time is expired, or

iii) a Multiplexing Indication has been detected.

The incoming frames have different frame lengths and their length has to be considered when they participate in the frame multiplexing. When a frame is received in the IQ buffer, its length is added to the current length of the frame which is under multiplexing. If the total length exceeds the maximum allowed length, the last received frame is rejected from the current formulation, a multiplexed frame is generated and a new multiplexing round starts. Otherwise, the frame is merged to the currently multiplexed frame, the 'participation indicator' of this frame is incremented by one, the new frame length is estimated, the multiplexed frame Routing Information field is updated and a new frame is expected to be received.

In order to satisfy the communication parameters, e.g. the 'quality of service', and to restrict the maximum allowed end-to-end delay, a maximum Waiting Time is defined. When a new multiplexing round starts, a timer is preset to a predefined Waiting Time. When this time expires, the multiplexing procedure is terminated and the next frame is expected to start a new round. This time is in close relation to the departure rate, in order to regulate the availability of frames in the TQ with the departure times.

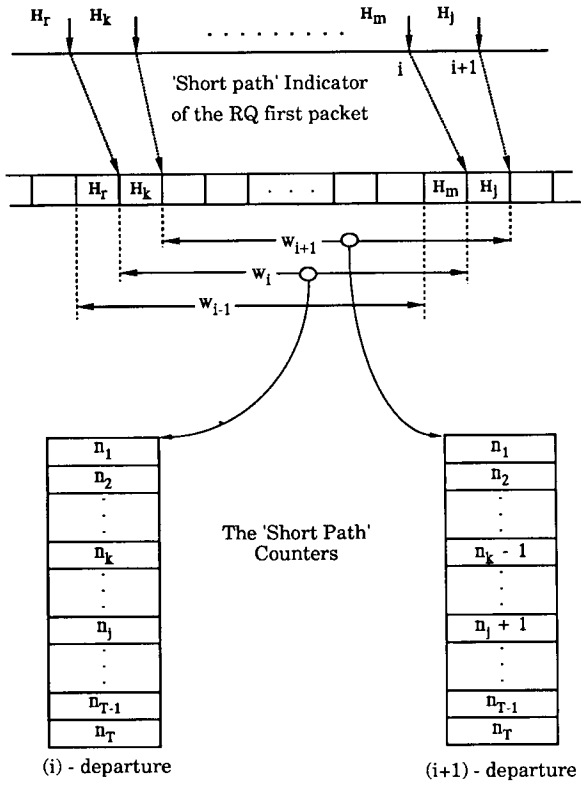

Fig. 4. The 'Statistic Window' method

As it was explained previously, the 'Statistic Window' Handler can be used to inform the bridge processor that an IQ buffer has to be allocated to a new Destination Counter. In this case, the 'Statistic Window' Handler provides the bridge number for which the new multiplexing procedure will begin and the bridge number for which the multiplexing procedure must be completed.

5. IMPLEMENTATION

In this section a conceptual design for the hardware implementation of the multiplexing mechanism is given and the architecture of a bridge with such an implementation is also described.

Fig. 5 shows the architecture of a bridge with two network interfaces and the hardware implementation of the Source Routing multiplexing algorithm, called Multiplexing Monitor Unit (MMU). The bridge is composed of the bridge processor, a common system memory for buffer descriptors exchange, a dual-port memory for frame exchange, two Multiplexing Monitor Units and two MAC interfaces. The Multiplexing Monitor Unit is a bus monitor unit which implements the previously described 'Statistic Window' during the frame reception procedure and it also uses the semantics of the proposed Source Routing multiplexing algorithm to determine the stages of the algorithm deployment. In the beginning of a frame, the MMU scans the frame to detect either the inclusion of Source Routing information or the existence of a multiplexed frame and keeps the result in its status register. If a Source Routing information has been detected, the MMU continues to scan the frame header to match one of the internal 'bridge-network-bridge' patterns. In the case where a 'broadcast' frame has been received, the MMU signals the bridge processor to transmit this frame with no multiplexing. After the frame transmission, the bridge processor can read the MMU indication registers and take information about the status of the last received frame. A more detailed description of the internal operation of the Multiplexing Monitor Unit is given at the last paragraph of this section.

For describing the bridge operation, we assume that the two MAC interfaces communicate with the bridge processor through common memory locations (in the system memory), while the received frames are stored directly in the dual-port memory. The basic organization of the buffer management for each interface is a circular

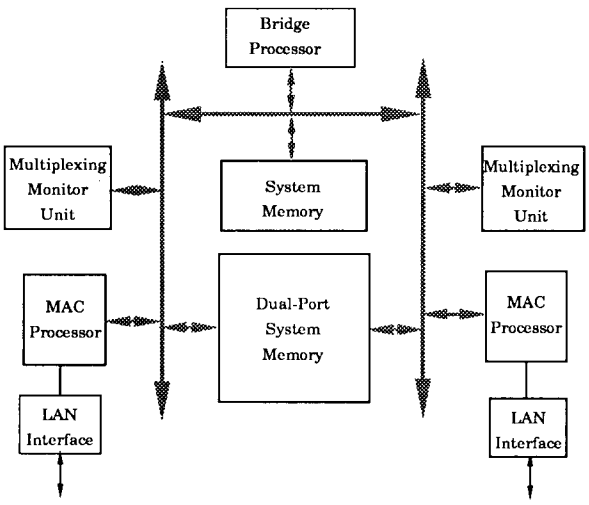

Fig. 5. The Bridge Architecture

queue of descriptors, relying in the common system memory, each descriptor holding a pointer to a buffer of the dual-port memory (starting address, length etc). The bridge processor and the network interfaces scan the respective descriptor rings to determine the next empty buffer in the receive mode, or the next ready buffer in the transmit mode.

The bridge processor manages the dual-port memory through the values given to the buffer descriptors. Each time the network interface receives a frame, stores it in the dual-port memory, updates its descriptor buffer and generates an interrupt to the bridge processor. Then the bridge processor reads the status and the indication registers of the MMU to determine the evolution of the existing multiplexing mechanisms or to start a new one. The bridge processes the buffer descriptors, handles the IQ buffers and, when a multiplexed frame is ready, passes the respective descriptors to the other network interface (TQ buffer), for transmission into the network. At the last stage of the multiplexing procedure, the bridge processor updates the required frame header and appends to it the stored data.

When the Multiplexing Monitor Unit detects that a multiplexed frame is received, it sets the respective bit to the status register. The bridge processor recognizes it when the frame reception is completed and reads the frame header to determine the number of the included frames and their positions in the dual-port memory. Using this information, the processor updates a number of buffer descriptors in the output network circular queue (demultiplexing procedure) and passes their control to the network interface.

This mechanism imposes no processing time delay to the frames traversing the bridge, because the multiplexing/demultiplexing procedures are executed during the frame reception. Another advantage of the proposed architecture is that there are no frame transfers inside the bridge for internal processing and the buffering delay is mainly due to the access protocols of the two networks. This architecture can be combined with the one proposed in [8], to give a high throughput bridge for Source Routing application in the high-speed interworking environment.

In Fig. 6, the block diagram of the Multiplexing Monitor Unit is given. The MMU is composed of six processing submodules activated sequentially as the algorithm is evolving. The CAM memory contains all the network $B_iS_jB_j$ patterns. When a frame is received, the Route Designators are compared with these patterns and if a match has been detected, the included priority encoder generates the address of the location of the pattern in the CAM. Each time a frame passes the CAM memory, it passes this address to the Traffic Window (a null address is passed if no

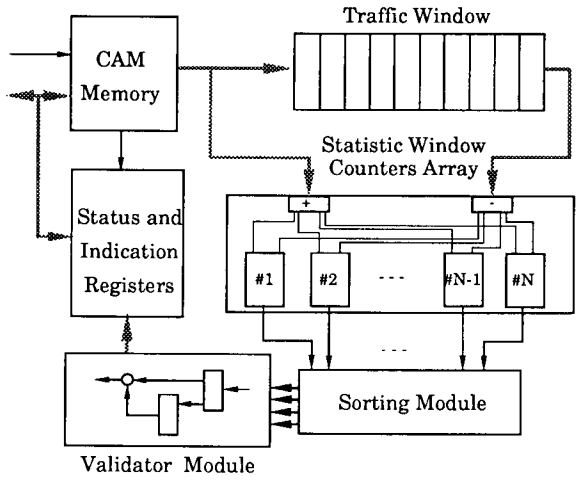

Fig. 6. The MMU Block Diagram

match is detected). The Traffic Window contains the traffic characteristics in encoded form and can be implemented using multiple parallel registers. When a new address is generated by the CAM memory, the first address of the Traffic Window is pushed out. The first and the last addresses of the Traffic Window are fed to the array of the Statistic Window Counters. The counter associated with the last address is incremented by 1, while the counter associated with the first address is decremented by 1. If the last address is the same with the first address, the counters' values do not change.

The new values of the array of the Statistic Window Counter feed the Sorting Module of the Multiplexing Monitor Unit. The Sorting Module compares these values and outputs the values of the four more active addresses in the current Traffic Window (for a four IQs multiplexing mechanism). The current results of the Sorting Module are compared with the previous results by the Validator Module and the status and indication registers of the MMU are updated. These new values will be used by the bridge processor to determine if there is any modification to the current buffer allocation and which are the parameters of the specific modification. The MMU can be implemented in a single chip due to its structure and it can be integrated with the most of the market available MAC processors.

CONCLUSIONS

The proposed extension of the Source Routing method is based on the traffic characteristics met in the Extended LAN environment, where at least one high speed network is attached. In such an environment, the use of the frame length limitations, imposed by the Source Routing method during the transmission in the high speed network, degrades the system performance and does not take into account the traffic conditions in the bandwidth allocation process. The proposed multiplexing method can be used in networks with long frame sizes and can improve the system performance.

The method uses the reserved bits of the Source Routing frame header and follows the existing frame format. The method improves the bandwidth allocation when used in a high speed backbone network, which interconnects various and probably incompatible bridges. The improvement is achieved by monitoring the traffic load in each bridge and allocating the internal buffers accordingly. The way the buffers are allocated is ideal for the frame multiplexing procedure and can be used to minimize the processing time overhead. This work also describes a conceptual design for the hardware implementation of the multiplexing method and the use of such an implementation in a high throughput bridge.

REFERENCES

[1] G. Varghese and R. Perlman: "Transparent Interconnection of Incompatible Local Area Networks Using Bridges", *IEEE Journal on Selected Areas in Communications*, Vol. SAC-8, No 1, January 1990, pp. 42-48.

[2] M. Soha and R. Perlman: "Comparison of Two LAN Bridge Approaches", *IEEE Network*, Vol. 2, No 1, Jan. 1988, pp. 37-43.

[3] F. Backes: "Transparent Bridges for Interconnection of IEEE 802 LANs", *IEEE Network*, Vol. 2, No 1, January 1988, pp. 5-9.

[4] R. C. Dixon and D. A. Pitt: "Addressing, Bridging and Source Routing", *IEEE Network*, Vol. 2, No 1, Jan. 1988, pp. 25-31.

[5] D. A. Pitt and J. L. Winkler: "Table-Free Bridging", *IEEE Journal on Selected Areas in Communications*, Vol. SAC-5, No 9, December 1987, pp. 1454-1461.

[6] D. Y. Lee and J. Y. Lee: "Performance Comparison of Bridge Algorithms in Interconnected Local Area Networks", *Computer Networks and ISDN Systems* 22 (1991) 265-276, North-Holland.

[7] J. Koutsonikos, T. Antonakopoulos and V. Makios: "A Dynamic Buffer Structure for LAN Gateways", *The 21st Annual IEEE Communicaton Theory Workshop*, Rhodes, Greece, June 1991.

[8] M. C. Hamner and G. R. Samsen: "Source Routing Bridge Implementation", *IEEE Network*, Vol. 2, No 1, Jan. 1988, pp. 33-36.