117

Architecture alternatives and performance issues in DQDB subnetwork interconnection

R. Dittmann, P. Tran-Gia, University of Würzburg (Würzburg, D)

Abstract

In this paper the performance of some alternative concepts to interconnect DQDB subnetworks is investigated. The aim is to give hints which architecture and load control strategy should be chosen for a given configuration. Especially, we devote attention to the performance comparison of bridge-level, slot-relay internetworking structures, where two DQDB subnetworks are taken into account. The approach discussed here corresponds to a Multiple Port Bridge structure operating with Derived MAC Protocol Data Units. We describe two alternatives: i) the headend-based and ii) the station-based internetworking unit architectures with different control complexities.

1 Introduction

The Distributed Queue Dual Bus (DQDB) access protocol has been accepted as a standard for the subnetwork of the IEEE 802.6 MAN (Metropolitan Area Network) in 1990 [4]. This first standardization step has been done mainly to specify a stand-alone DQDB architecture. Thus the next logical step is to consider interconnectivity capabilities between DQDB subnetworks, between DQDB and WAN (Wide Area Network) such as ATM (networks operating in Asynchronous Transfer Mode) as well as DQDB used as access networks (GFC: Generic Flow Control) in high-capacity systems [1, 10, 6]. These issues are also currently under study by the IEEE 802.6 Working Group and the CCITT Working Group XVIII.

In this paper we devote attention to the interconnection of DQDB subnetworks, where performance aspects will be particularly taken into account. We will start with an brief description of the DQDB access protocol and subsequently discuss some architectural alternatives in connecting two DQDB subnetworks. An example will be taken and impacts of the network architecture on the system performance will be shown in conjunction with simulation results.

Detailed descriptions of DQDB access protocol and systems can be found in [4]. There is a number of simulation studies [6, 8, 12] and approximate analyses for statistical traffic [2, 6, 11] considering performance aspects of various successive releases of the standard proposals. The structure of a DQDB access system is depicted in Fig. 1 showing a stand-alone network. The bus system, which operates in a slot-based manner, consists of a pair of slotted unidirectional buses flowing in reverse directions. A station attached to the dual bus system observes data passing by on the two buses and uses the bus system according to a distributed queueing scheme applied to the global system.

The access mechanism is identical for the two buses. Hence, we will describe the access mechanism only for one direction. A slot contains an access control field (ACF), a segment header and a segment payload area for isochronous or asynchronous traffic. For these two different types of traffic two access control modes are provided. The pre-arbitrated access mode is reserved for isochronous services like voice and CBR video. Accesses of non-isochronous services are controlled by the station itself according to the queued-arbitrated [4] access mode.

If a station has to transfer a data segment downstream using bus A, it notifies this wish to all stations upstream by sending a request on bus B, i.e. setting a request bit. The station continuously counts all requests flowing by on bus B. The station schedules only one segment per bus, which has to wait until all preceding requests observed on bus B are served before it is transmitted.

Considering again data transfer on bus A, a station can be in the following two states: IDLE and COUNTDOWN. For each bus the station has to maintain mainly two counters, the request counter (REQ_CNT) and the countdown counter (CD_CNT) (see Fig. 1).

1. IDLE-state: the station has nothing to send or was on immediate transition from state COUNTDOWN. The request counter is decremented upon observing a free slot flowing by on bus A and is incremented upon seeing a request passing by on bus B.
2. COUNTDOWN-state: the station has data segments to transmit. A segment has been scheduled for transmission.

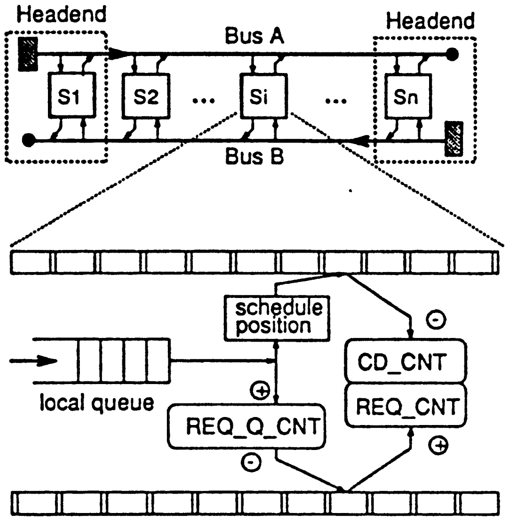

Figure 1: *Stand-alone DQDB network*

Papers presented at EFOC/LAN '92, the Tenth Annual European Optical Communications and Networks Conference, held in Paris, France, June 24–26, 1992.
Sponsored by IGI Europe, Inc., a Division of Information Gatekeepers, Inc., 214 Harvard Avenue, Boston, Massachusetts 02134 USA.
Telephone (617) 232-3111 : Telefax (617) 734-8562.

In this state, the countdown counter is decremented by observing a free slot flowing by on bus A while the request counter is incremented upon arrival of a new request on bus B.

3. State transitions: A state transition from IDLE to COUNTDOWN is processed as follows. The station enqueues a request to the local request queue (which is implemented by the request queue counter REQ_Q_CNT), sets the countdown counter to the actual value of the request counter and then resets the request counter. The station always takes over from COUNTDOWN to IDLE after sending a segment. This is followed immediately by a backward state transition from IDLE to COUNTDOWN if there are still segments waiting in the station.

2 Architecture alternatives

2.1 General considerations

We consider the following migration scenario, starting with an existing DQDB stand-alone network. Due to an increasing number of stations to be connected or to larger geographical network environment, the existing network has to be extended, where a number of additional subnetworks should be developed, installed and interconnected. These extension networks can be of a higher speed in accordance to a later technology. The architecture and performance problems arising along this migration path will be discussed in this subsection. In current standard proposals this is included in the scope of "Multiple Port Bridging of Metropolitan Area Networks" [5].

In general, the two subnetworks can be interconnected 1) by means of a direct connection on physical layer, 2) using a backbone MAN or WAN or 3) through an Internetworking Unit (IU) on one of the lower protocol layers.

The most simple way to interconnect the two DQDB-subnetworks is to build a larger network by connecting the two buses in a physical sense according to solution 1), if necessary by means of a repeater. The local traffic of one subnetwork has to flow through the other and probably obstructs the local traffic there. From performance viewpoint it is obvious that this solution is quite disadvantageous. For a large network infrastructure solution 2) is a preferred one. This needs a higher interface complexity in terms of switching functions, buffering and resource allocation problems. A similar solution can be observed in the current definition of access networks such as in GFC proposals. Although architectures of type 2) are as well interesting for traffic matrices with intense local subnetwork components, we will restrict ourselves in this paper to performance comparison of solutions of type 3).

In general, an Internetworking Unit operating at DLC- or higher protocol layers, i.e. with a physical separation, forwards only the internetwork traffic part to the other subnetwork. From topological viewpoint we will discuss and compare the following two types of Internetworking Units (IU): 3a) Headend-based (HB), where the IU connects two headends or headend stations and 3b) Station-based (SB), where the IU can be located at any position along the bus (cf. Fig. 2). From flow control viewpoint we will distinguish two cases i) the IU does not use the request streams on the reverse bus and ii) uses them to perform a more efficient load control.

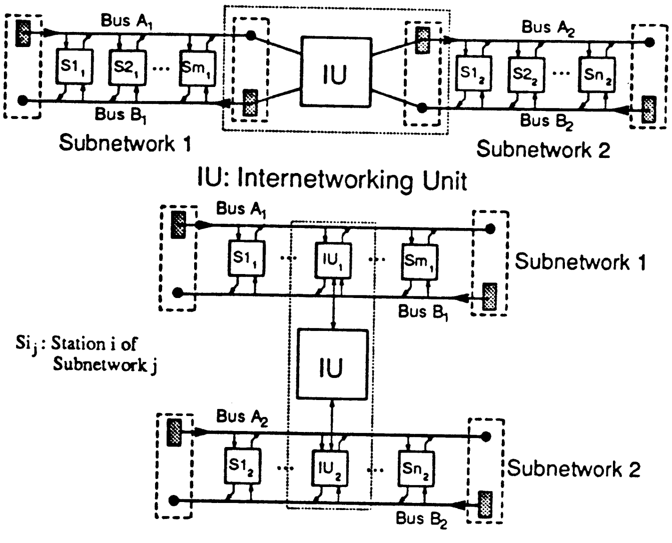

Figure 2: *Headend- and station-based interconnection*

Since the operation speeds are expected to be high for future DQDB networks, for larger volume of inter-subnetwork traffic, solutions on high protocol layers (4 to 7) seem to be very time- and overhead-consuming.

According to the DQDB standard [4] the MAC (Medium Access Control) layer deals with two types of Protocol Data Units: i) the Initial MAC Protocol Data Units (IMPDU) and ii) the Derived MAC Protocol Data Units (DMPDU). The IMPDUs represent messages to be transmitted while the DMPDUs correspond to the segmented data units contained as payload part in slots. Thus, from internetworking viewpoint, IMPDUs should be treated in a end-to-end manner while DMPDUs should be handled link-by-link. An Internetworking Unit working at a layer above the MAC-layer has to reassemble IMPDUs before it passes them on. From performance viewpoint it is obvious that this architecture will lead to larger transfer delay compared to a MAC-layer solution using DMPDUs as information units [9]. In the case of internetworking with other subnetwork types, e.g. IEEE 802.3-5, FDDI etc., bridging may have to be done in dealing with IMPDUs. In a network consisting only of DQDB subnetworks, however, bridging can be done in a more efficient way using DMPDUs (i.e. segments) as data units. This slot-based DQDB internetworking approach will be discussed in this paper; this can be seen as a slot-relay internetworking approach.

It should be mentioned here that the slotted medium of the DQDB protocol provides additionally facilities to make the internetworking function more efficient. Since slots carring outgoing internetwork traffic are no longer relevant for local downstream stations, the Internetworking Unit processing DMPDUs could erase them. Thus, the IU may reuse erased slots to serve traffic of the other subnetwork. Clearly, in doing this, the IU is able to pass on internetwork traffic without increasing the load of the connected subnetworks.

Due to the interconnection, we have to consider the phenomenon that the subnetworks could overload each other via the IU. We will address briefly the control problem of internetwork traffic. In high-speed networks control information is generally received with an non-neglectable delay due to the large propagation delay compared to slot duration. Therefore flow control should be done in an end-to-end fashion at a higher protocol layer, i.e. rather with messages than slots as information unit.

On the other hand overload caused by internetwork traffic affects a subnetwork already at the MAC-layer. When the bandwidth-balancing mode is active, the IU is not able to dominate the subnetwork as a heavy user. Due to the transient phase of the bandwidth-balancing mode the local traffic could still be obstructed, especially if this consists mainly of short messages [12]. During a high-load phase the buffer in the IU can dramatically be overloaded. This motivates the introduction of a congestion control scheme at the MAC-layer. In the case of subnetworks with different speeds a load control method becomes mandatory to save the slower network from being flooded by internetwork traffic. This function can e.g. be implemented using the DQDB control facilities on the MAC-layer by putting requests into the other subnetwork. This mechanism should carefully be used since these additional requests can throttle down all remote stations and reduce the bus system utilization. Some alternatives for this way to regulate the load caused by internetwork traffic will be discussed in detail in subsections 2.2 and 2.3.

In the later sections we will consider the interconnection of two DQDB systems: Subnetwork 1 with stations $S1_1, \ldots, Sm_1$ and Subnetwork 2 with stations $S1_2, \ldots, Sn_2$ (cf. Fig.2). The Internetworking Unit is mainly a bridge operating at the MAC-layer using DMPDUs. Since the operation of the bridge is symmetrical, below we will only consider the traffic flowing from Subnetwork 1 to Subnetwork 2 in the description of the bridge functionality. Furthermore we assume that the bridge does not create traffic itself. In the following we will describe the two alternatives: headend-based (HB) and station-based (SB) bridge architectures in detail.

2.2 Headend-Based Bridge

In a network environment with several DQDB subnetworks having their headends in the same geographical location, e.g. in the same building, the same computing center etc., the obvious way to connect pairs of DQDB-subnetworks is to use a headend-based bridge (HB-bridge). It should be noted here that the same topology with HB-bridge will arise if we divide a large network with communities of interest (e.g. departments, groups of neighboring users,..) into interconnected smaller subnetworks. This is a logical alternative to the single network architecture with heavy overall superimposed traffic. The operation of this bridge is simpler than in case of station-based bridge, since here, arriving data traffic is sent on only at one bus. The PSR-bit in the slot header can e.g. be used to recognize and determine internetwork traffic, because all pure local data is already read before reaching the headend. Two HB-bridge type will be considered:

1. In general a headend-based bridge is a special station-based architecture connecting the last station of Subnetwork1 with the first one of Subnetwork2 (*HB-bridge type I*). We will discuss it in detail in the next subsection.

2. With both subnetworks running at the same speed, we can construct a headend-bridge similar to an erasure node (*HB-bridge type II*), i.e. it does not generate free slots like a usual headend, but erases all local traffic of the subnetworks crossing the bridge. The possibility to connect DQDB subnetworks via architectures similar to erasure nodes like the HB-bridge type II and the necessity to compare this with other architectures are mentioned in [7]. Here we need just enough buffer space and processing time to be able to read the PSR-bit of the subsequent slot and to compute the possible changes of the slot header. If there is pre-arbitrated local traffic to be transmitted, additional buffer is needed to prevent interference with internetwork traffic.

With the HB-bridge type II, a subnetwork can be temporarily overloaded and flooded by internetwork traffic. This phenomenon would obstructs the local traffic heavily. Since this effect is expected to occur rarely and to be of short-term, it could be managed without additional control mechanism. On the other hand there might be obstruction for a considerable period, when e.g. file servers or other rather bursty sources are concentrated in one subnetwork. Therefore an optional ability to regulate the load flowing through the IU, caused by internetwork traffic, might be helpful. One possibility is to allow "internetwork requests". We consider e.g. the case where the Subnetwork 2 is in an overload situation. Recognizing the overload condition, the IU can reserve slots in Subnetwork 1 to slow down the traffic in this subnetwork. This mechanism will help to throttle the internetwork traffic flowing to the overloaded subnetwork during the critical interval. It prevents further a possible starvation of local traffic in a subnetwork in a period, during which the incoming internetwork traffic dominates the bus system.

In principle two network configurations can be thought of:

1. There is a single large network with one erasure-node and the bridge processes requests according to a dedicated protocol.

2. The two subnetworks are separated and the bridge creates only internetwork requests if it recognizes overload conditions.

In the first configuration the bridge has to keep the information concerning the distributed queue up-to-date. Therefore the number of requests has to be reduced according to the number of erased slots. A possible method to do this is to cancel requests as proposed in [13]. Here we must be sure that any downstream station can use the erased slot before a request-bit is reset. Due to propagation delay some requests may survive, although they are not up-to-date any more. This slightly decreases the available bandwidth for upstream stations. On the other hand no hardware changes at other stations are required as by more sophisticated protocols using "negative requests" [3], "handshake requests" [10] or marking erased slots, thus downstream stations only produce requests if necessary [1].

For the second configuration we have to recognize overload conditions caused by internetwork traffic. For that purpose the bridge has a request-counter, which is handled according to the DQDB protocol. If there is no internetwork traffic its value is zero or one, since the bridge is able to decrement it with the same rate as request may arrive. The value of this counter increases with the intensity of the internetwork traffic. We can assume overload if the request-counter is greater than zero for a certain period of time, which may depend on its value. It is also possible to monitor the internetwork traffic and assume overload, if its bandwidth exceeds a threshold while the request-counter is greater than zero.

If the bridge recognizes overload conditions caused by internetwork traffic, it sets request-bits of slots created for the other network. The number of these slots can be derived from the current value of the request-counter, e.g. this value itself, or estimated dynamically by sending requests until this value falls below a threshold. Using requests at a currently unused higher priority level improves the efficiency of such a control mechanism, because they affect the countdown-counter of active stations at the other subnetwork.

2.3 Station-Based Bridge

In the case of connecting two given subnetworks the location of the IU is chosen according to certain criteria, e.g. the shortest distance between subnetworks. Therefore it may be necessary or advantageous to have a bridge architecture which connects the subnetworks somewhere along the bus rather than at the headends. A station-based bridge (SB-bridge) can generally be constructed and modeled as a pair of stations, which has to process the whole incoming internetwork traffic. The SB-bridge has to test the destination address of each slot crossing it on a subnetwork. Unless the bridge is located at a headend station, the PSR-bit cannot be used here, since unread local traffic streams can pass the bridge. If a slot carries internetwork traffic, the bridge enqueues the data segment (DMPDU) in its local buffer dedicated for the other subnetwork. Thereafter it operates according the distributed queue protocol or to a variant of it.

Several protocol variants can be implemented at the bridge, e.g.:

1. The bridge acts strictly according the DQDB protocol, thus it is not able to erase slots.
2. The bridge is allowed to erase slots and operates like a normal station, which has received a free slot. Here an erasure-node protocol should be implemented to avoid misranking in the global distributed queue.
3. The bridge is allowed to replace data segments of slots carrying outgoing internetwork or already read local traffic. Thus the bandwidth for incoming internetwork traffic has been increased without confusion about the status of the distributed queue due to additional free slots.

A station-based bridge may also be allowed to create internetwork requests. In contrast to the architecture similar to erasure-nodes, a subnetwork cannot be flooded by internetwork traffic, since its datasegments are queued in the local buffer of the bridge. Therefore the bandwidth-balancing mode will save the local traffic from considerable obstruction in most cases. On the other hand there exist internetwork traffic streams from downstream and upstream stations; in total it can have a higher intensity than by a single bus. Thus, this bridge type needs the facility to create internetwork requests to prevent overflow of its buffers. Here the queue length can be viewed as an indicator for overload caused by internetwork traffic.

The interconnection of subnetworks with different speeds causes some particular problems. In this case the internetwork traffic processes are expected to be more bursty and their intensities more non-symmetrical. The faster network is able to overload the slower one with its internetwork traffic. Hence, for these system configurations larger buffers and efficient congestion control mechanisms are required. It may be possible, e.g. to give the bridge station of the slower subnetwork a higher bandwidth balancing modulus. Thus it can get more bandwidth than other stations of this subnetwork in high load phases (cf. [9]) to prevent a buffer overflow before mechanisms at a higher layers are able to react to the overload.

2.4 Architectural comparison

The main difference between headend-bridges of types I and II is that a bridge of type I sends on the internetwork traffic according to a relay function without paying attention to particular protocol mechanisms in the subnetworks. Implicitly this traffic has a higher priority as the local traffic. This results in shorter end-to-end delay for internetwork traffic at the cost of larger delays for the local traffic. We could get similar results for other bridge types in using a currently unused higher priority for transmitting internetwork traffic and disabling the bandwidth balancing mode at the bridge.

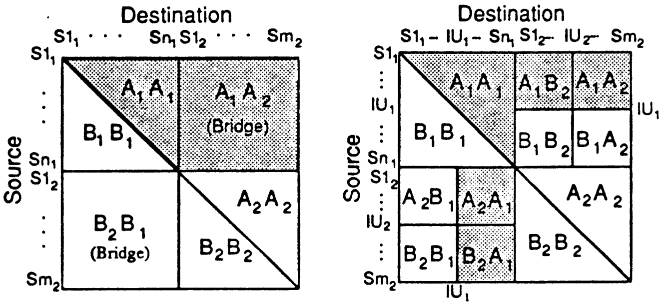

Figure 3: *Traffic streams for Headend-based and Station-based architectures*

In the following we compare architectures to connect two subnetworks at headend stations and in the midst of their buses respectively. For that purpose we take a closer look on the traffic flows on a particular bus, e.g. bus A1 marked in Fig. 3. A HB-bridge has upstream stations only on the buses A1 and B2. The internetwork traffic only flows from A1 to A2 and from B2 to B1, thus it can arrive and be forwarded with the single bus speed. In contrast a station-based bridge in the middle of the bus has upstream stations on each bus. The internetwork traffic to bus A1 could arrive at both buses A2 and B2. Hence, the bridge temporarily might have to deal with arrivals at twice the bus speed. On the other hand such a bridge is able to share the internetwork traffic coming from bus A1 among the buses A2 and B2, that means it can use the capacity of both buses to serve this traffic. Furthermore the performance can be improved by slot erasure only for station-based architectures.

Therefore the performance of a bridge is strongly influenced by the traffic matrix and the locality of the traffic streams. If there is a concentration of stations, e.g. fileservers, at the end of a subnetwork and these stations send large messages to all stations of the other subnetwork, a connection in the midst of the buses is the preferred one to share the load among the buses. In the case of a subnet with a station, e.g. a gateway to a WAN, which receives large messages from all stations at the other subnetwork, a headend-based architecture is expected to perform better, since a bridge in the midst of the bus might be temporarily overloaded from the superimposed internetwork traffic of both buses.

In case of a heterogeneous environment with subnetworks operating at different speeds a combined solution would be suggested. The bridge should connect the faster subnetwork at a headend to the slower one at the midst of its bus system. Thus the internetwork traffic coming from the faster one can only arrive with the bus speed of this subnetwork, but might be shared among the buses and therefore served with the double speed of the slower one. In the opposite direction the bridge might be able to use the higher capacity of the faster subnetwork by offering superimposed internetwork traffic of both buses.

3 Performance of a DQDB interconnection example

In this section simulation results will be presented to compare the bridge architecture discussed in the previous subsections. We take the interconnection of two subnetworks into account, each with a bus length of 11km (i.e. a logical bus length of 16 slots by bus capacity of 150 Mbps) and 9 equidistantly located stations. Each station generates the same total traffic amount. We assume the input processes with geometrically distributed batch size of 10 slots mean; the batch interarrival time is negative-exponentially distributed.

To indicate the intensity of the internetwork traffic, we introduce a locality factor, which is defined to be the ratio between the total local traffic and the total traffic volume of all the stations.

Fig. 4 shows the breakdown of the end-to-end packet delay, where its components (waiting time in the local queue and head-of-queue delay) are drawn. Here a comparison of the HB-bridge type II ($S9_1$ and $S1_2$) and the SB-bridge in the midst of both bus systems ($S5_1$ and $S5_2$) is depicted, the SB-bridge operating strictly according to the DQDB protocol. The total amount of local traffic is $\rho_{local} = 0.5$ and the intensity of internetworking traffic is $\rho_{internetwork} = 0.3$ at each bus. Thus, the locality factor is here 0.625. The station-dependency of the end-to-end delay due to the propagation delay is conspicuous for the HB-bridge. Since traffic arrives slot by slot at the bridges, at a bridge station the waiting time in the local queue is much smaller than at the other stations with batch arrival.

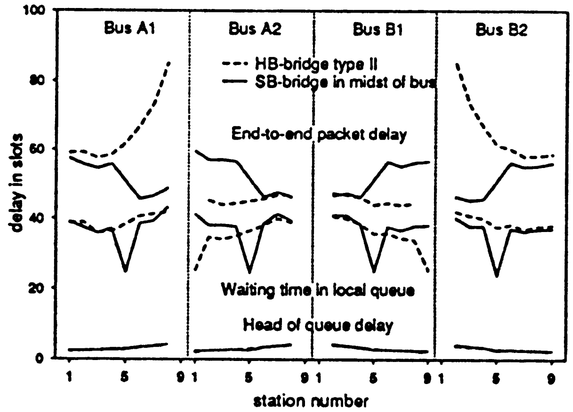

Figure 4: *Delay comparison of HB- and SB-bridge*

A comparison of SB-bridge structures is given in Fig. 5, where the impact of protocol variants (as discussed in Subsection 2.3) is shown. Taking into account only the waiting time in the local queue, the best performance can be observed for the variant with slot erasure. Here we use a simple extension to the DQDB-protocol: every time the bridge has erased a slot, it decrements its request queue counter if it is greater than zero, otherwise the bridge increments a additional counter called Slot-Erased-Counter, which has an upper limit (e.g. a bus latency). If a request arrives at the bridge while this counter is greater than zero, the request is canceled and the counter is decremented. With a more sophisticated protocol variant we might obtain even better results at the cost of additional hardware.

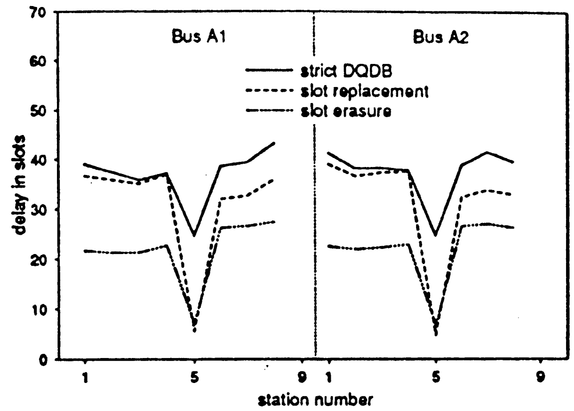

Figure 5: *Delay comparison of different SB-bridge architectures*

Under heavy traffic conditions the advantage of the erasure protocol variant of the SB-bridge becomes remarkable, as depicted in Fig. 6, where a comparison of the end-to-end packet delay for the two traffic intensities $\rho = 0.5$ and $\rho = 0.9$ is made, while the locality factor is 0.625 in both cases. It can be observed in this Figure that delay balancing is obtained by employing the variant with slot erasure function.

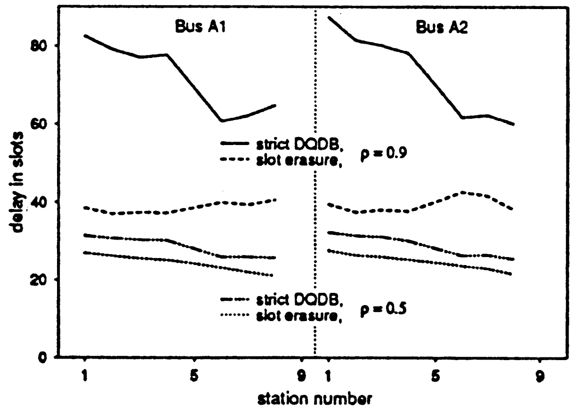

Figure 6: *Heavy traffic behavior*

As expected the delay in the SB-bridge solution is strongly affected by the locality factor of the traffic matrix. This phenomenon is the subject of the comparison of the waiting times in Fig. 7. Here the bridges operate strictly according the DQDB protocol and the total amount of traffic is $\rho = 0.8$ for each bus in all cases.

In Fig. 8 an environment with two DQDB subnetworks operating at different speeds (the bus system A1/B1 is twice as fast as the bus system A2/B2) is considered. It can be seen that the headend-based bridge solution for both bus systems is disadvantageous. The solution proposed in subsection 2.4 shows slightly better behavior than a SB-bridge at the midst of both bus systems.

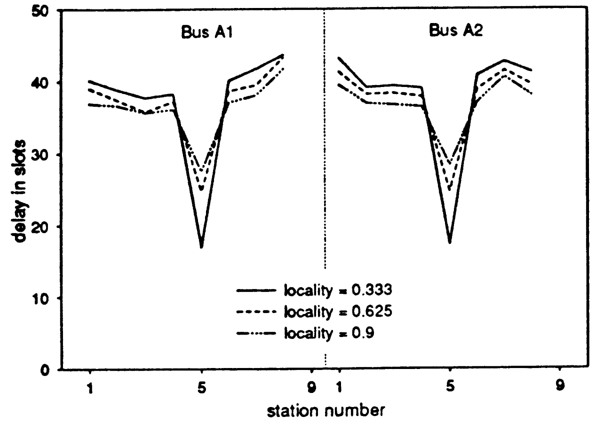

Figure 7: *Influence of traffic locality*

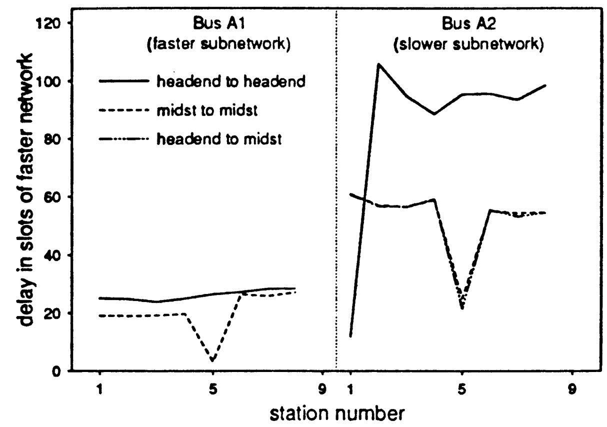

Figure 8: *Effect of network heterogeneity;* $c_1/c_2 = 2$

4 Conclusion and Outlook

We discussed performance issues in conjunction with some alternative concepts to interconnect DQDB subnetworks. The study focused on DQDB-to-DQDB bridge architectures where the Derived MAC Protocol Data Unit (DMPDU) was considered to be the basic information unit. There are various reasons for this bridge type. On the one hand the slot-based bridge architecture could be used to support message-based bridge architectures. On the other hand using slot-based architecture real high-speed bridges can be designed to interconnect DQDB subnetworks, due to the simple bridge function and the overload control capability. Two alternatives have been described: i) the headend-based and ii) the station-based bridge architectures with different control complexities. Using simulations, access and transfer delays of these configurations have been compared, where the basic case of connecting two DQDB subnetworks was taken into account. The results obtained show that in term of delays the bridge performance is very sensitive to the traffic relationship between stations and the location on the buses.

Acknowledgement
The authors would like to thank T. Stock for the discussions during the course of this work and for reviewing the manuscript.

References

[1] Australian Telecom, "A Generic Flow Control Protocol for B-ISDN", CCITT Study Group XVIII, Contribution No. D.1660, Melbourne, December, 1991 (see also Budrikis, Z.L., "Australia Backed Proposal of GFC Protocol", T1S1 Document T1S1.5/91-393, November 1991).

[2] Conti M., Giordano S., Gregori E., Lenzini L., "DQDB Modeling: Problem complexity reduction and solution via Markov Chain", Int. Conference on the Performance of Distributed Systems and Integrated Communication Networks, Kyoto, Japan, September 1991.

[3] Garett M.W., Li. S.Q., "A study of slot reuse in dual bus multiple access networks", Proc. IEEE INFOCOM '90, San Francisco, 1990, pp. 617–629.

[4] IEEE 802.6 Working Group, "Distributed Queue Dual Bus (DQDB) Metropolitan Area Network (MAN)", IEEE Standard 802.6, December 1990.

[5] IEEE 802.6A Subworking Group, "Proposed Standard: Multiple Port Bridging of Metropolitan Area Networks (MANs)", IEEE P802.6a/D0.1, September 1990.

[6] Mukherjee B., "A journey through the DQDB network literature", IBM Research Report, RC 17016, July 1991.

[7] Rodrigues M.A. "Erasure node: performance improvements for the IEEE 802.6 MAN", Proc. IEEE INFOCOM '90, San Francisco, 1990, pp. 636–643.

[8] Sauer K., Schödl W., "Performance Aspects of the DQDB Protocol", Proc. ITC Specialist Seminar, Adelaide, Sept. 1989, also in Computer Networks and ISDN Systems, 20 (1990) 253–260.

[9] M.P. Spratt, "Bandwidth Allocation over Several Networks with DQDB", EFOC/LAN 1991, London, June 1991.

[10] SWISS-PTT, "The SWISS-PTT GFC protocol; New Features, in Response to Questions Raised in Geneva, June 1991", CCITT Study Group XVIII, Contribution No. D.1684, Melbourne, December 1991, (see also PRO TELECOM/ASCOM, "Ascom GFC Protocol Proposal: Status after the Melbourne CCITT SG XVIII Meeting", ETSI-NA5 TD 92/89, Montpellier, France, January 1992).

[11] Tran-Gia P., Stock T., "Approximate Performance Analysis of the DQDB Access Protocol", Proc. ITC Specialist Seminar, Adelaide, Sept. 1989, also in Computer Networks and ISDN Systems, 20 (1990) 231–240.

[12] van As H.R., "Performance Evaluation of Bandwidth Balancing in the DQDB MAC Protocol", Proc. 8th EFOC/LAN '90, Munich, F.R.G., June 1990.

[13] Zukerman M., Potter P., "A Protocol for Eraser Node Implementation within the DQDB Framework", Proc. GLOBECOM '90, San Diego, December 1990.