The DTM Gigabit Network

Christer Bohm[1], Per Lindgren[1], Lars Ramfelt[1], Peter Sjödin[2]

Abstract

DTM is a fiber-optic network, based on bandwidth reservation and with support for dynamic reallocation of bandwidth. It is designed for real-time multimedia applications and for high-speed computer communication. DTM uses a novel medium-access technique and provides a multicast connection-oriented service. Several DTM networks can be connected into one large network. A prototype implementation and testbed is being constructed.

1 Introduction

Fiber-optic networks running at several gigabits per second will be the main information carriers in the future. They will replace the current networks for CATV, telephone traffic and data communication and integrate the functions of these into one network. Also, high-speed fiber-optic networks will have new important areas of applications, such as distributed multimedia with audio and video communication. MultiG is a Swedish research program focusing on distributed multimedia applications, and on how they can be supported by high-speed optical networks [15].

Distributed multimedia applications impose many requirements on networks. First, the network should have sufficient capacity to carry high-quality video and audio signals. It should have short response time to support person-to-person communication with audio and video. These applications also require timely delivery of data. For instance, video images may have to be presented on video screens at regular intervals. Finally, the network should support broadcasting so that one sender can efficiently transmit to many receivers.

DTM is a networking project within the MultiG program. It is a project to design and build a prototype of a fiber-optic high-speed network. This network is based on a new medium access technique—the Dynamic synchronous Transfer Mode, from which the project has its name. The DTM project further includes development of protocols, network resource management and network architecture. We will use the term “DTM network”, or just “DTM”, to refer to all these components in the project.

1. Department of Teleinformatics, Royal Institute of Technology, Stockholm, Sweden, {bohm,perl,lars-h}@it.kth.se

2. SICS, Box 1263, 16428 Kista, Sweden, peter@sics.se

This paper describes and motivates the DTM design, and the prototype DTM network. A description of an earlier version of DTM can be found in [4].

1.1 Design Goals

The DTM medium access technique is an effort to combine the advantages of asynchronous and synchronous media access schemes. It is fundamentally a Time Division Multiplexing (TDM) scheme, and as other such schemes it guarantees each host a certain bandwidth, and uses a large fraction of the available bandwidth for effective data transfer. In addition, the DTM scheme has in common with asynchronous schemes, such as ATM [9], support for dynamic reallocation of bandwidth between hosts. This means that the network can adapt to variations in the traffic, and divide its bandwidth between hosts according to their demand.

The service provided by a DTM network is based on the *multicast channel* abstraction. Hosts connected to a DTM network communicate with each other on channels. To each channel network resources are associated. These resources are given to a channel when it is created, and they remain constant throughout the channel's lifetime. This implies that channels provide services with guarantees on real-time properties. In particular, a channel has constant throughput and delay.

Channels are multicast, meaning that any number of hosts can communicate on a channel. Senders and receivers can be associated with channels independently of each other, and independently of the allocation of network resources to channels. For instance, when a broadcast channel has been created, hosts in the network can, at any time and at their own will, "tune in" and start receiving the data broadcast on the channel.

The intention is that channels should be so flexible that they can be used for most types of communications. For example, channels can be used efficiently for broadcasting, for point-to-point connections and for sending datagrams.

1.2 Disposition

The outline of this paper is as follows. The next section introduces the DTM medium access technique and protocols, and describes the main ideas in the design. Section 3 describes how DTM networks are combined into larger networks, and explains how switching is done between them. In Section 4, the prototype is described and status of ongoing work is reported. Section 5 makes some comparisons with related work and section 6 concludes the paper and outlines further work.

2 DTM Overview

The DTM network was designed taking into account how it could be built in hardware, as well as what its applications would be. The results of these considerations are as follows.

- DTM high-speed electronics can be built with few and standard components. This makes the network robust and reduces hardware costs.
- DTM can be implemented as an *all-optical* network, that is, a network where electro-optical conversion occurs only at the sender and the receiver [7]. Such networks are tolerant to node failures, have low probability of bit errors, and have low propagation delay.
- A DTM network provides services with real-time guarantees, in particular guarantees on throughput, access delay (the time it takes to setup a service), and propagation delay (the time it takes to transfer data from sender to receiver).
- DTM supports broadcasting, both for distributed multimedia applications such as lectures in a distributed university, as well as for more traditional video and audio broadcasting.
- There is no error handling in DTM. Different applications have different tolerance to transmission errors, and require different error-handling policies. Therefore we leave error-handling to application, specific higher-level protocols.

There is, however, a trade-off between throughput and access delay in DTM—it can take long time to create channels with a lot of bandwidth. We justify this by claiming that applications rarely have high demands on both throughput and delay. For example, an application that requires high throughput and low propagation delay, such as two-way video communication, probably tolerates long access delay. Also, applications that need low access and propagation delay, such as remote procedure calls, probably have modest demands on throughput.

#include “/afs/cell/home/sics/peters/DTM-demo/Results/Throughput8.ps”

2.1 Network Structure

A DTM network consists of *dual buses* which connect a set of *nodes*, as illustrated in Fig. 1. A dual bus is a pair of optical fibers where each fiber is used for data transmission in one direction. For example, in Fig. 1 a node sends data on fiber A to nodes with higher indexes, and on fiber B to nodes with lower indexes.

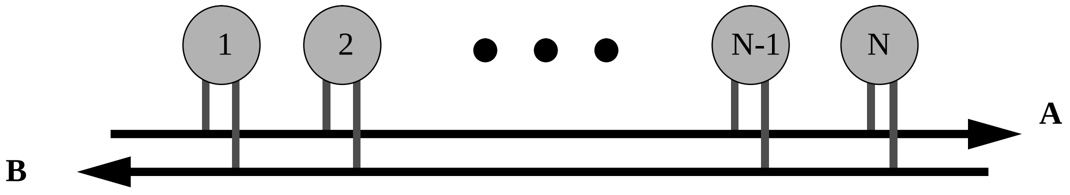

FIGURE 1. A DTM network with N nodes

Several *hosts* can be connected to each node. Hosts can be super computers, multimedia workstations, HDTV cameras and monitors, and local area networks, in which case the DTM network serves as a backbone network.

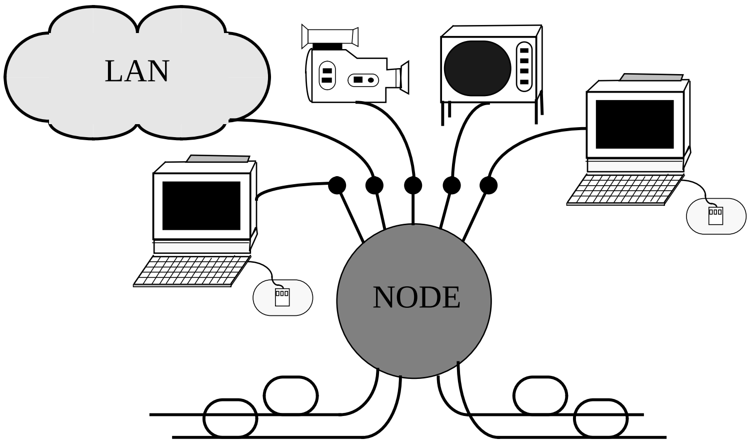

FIGURE 2. Example of a DTM node connected to one dual fiber bus.

Electro-optical converters represent a large part of the total hardware cost for fiber-optical networks. We decided to design DTM so that hosts can be connected indirectly (via nodes) to the fiber, which is done for instance also in the Hangman network [22]. This architecture allows several hosts to share interface hardware. It also reduces the number of units connected to the network, which simplifies network management. DTM does not rule out the alternative solution—to attach hosts directly to the medium—and when electro-optical converters become cheap this may be the preferred architecture.

The DTM protocols and medium access technique can, in principle, be used on any type of network. Besides the dual bus, a ring and a folded bus were considered. The dual bus was chosen mainly for the following reasons:

- It has the lowest average distance between two nodes.
- Synchronization and framing is simpler on a dual bus than on a ring.
- It has inherent broadcast capabilities.

2.2 The Time Division Scheme

Nodes access fibers according to a Time Division Multiplexing (TDM) scheme. The bit stream on a fiber is organized into *slots* where each slot can carry 64 bits. Slots are grouped into 125 µs long *cycles*, so the number of slots in a cycle is

$$N = B \cdot \frac{125 \cdot 10^{-6}}{64}$$

where B is the bit rate on the fiber. In the first prototype with 622 Mbit/s SONET OC-12 optics, there are thus about 1200 slots per cycle.

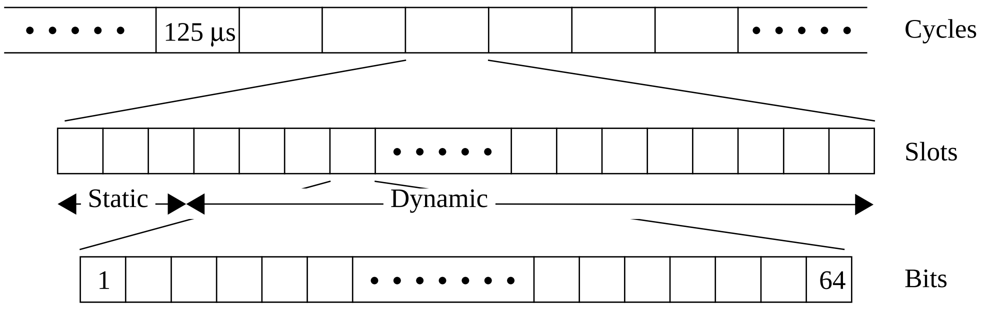

FIGURE 3. DTM Cycle Structure

A DTM fiber can be thought of as a flow of contiguous slots. A node on the fiber must pass this flow, slot by slot, from the incoming to the outgoing side of the fiber. A node can do two things with each slot that passes. First, the node can read the data in the slot, and give it to a host or another fiber. Second, the node can take data from a host or another fiber and write it into the slot. The DTM *fiber access protocol* ensures that a node does not write data into slots that already carry data. A node does not have to read a slot to see if it is free to use. Likewise, a node does not have to read a slot to decide what data to write into it. We claim that this absence of *read-modify-write* operations in DTM simplifies fiber access hardware, and makes it possible to use the DTM protocol on all-optical networks.

The decision not to use read-modify-write operations imposes restrictions on the DTM fiber access protocol. In DQDB [13], for instance, each cell contains a *busy* flag indicating whether the cell is currently in use. A node may write data into a cell only if its busy flag is cleared. To do this, a node must remove the cell from the fiber and examine the flag. If it is cleared, the node may put the cell back on the fiber with a new content, otherwise it must put the cell back intact. This procedure violates the DTM design decisions, so we do not use a similar *busy* flag in DTM slots.

The DTM fiber access protocol is based on the idea that there are two kinds of slots in a cycle, *static* slots and *dynamic* slots, as shown in Fig. 3. The static slots are for sending control information between nodes while the dynamic slots represent payload, i.e., they carry data. A slot is always owned by exactly one node, and only the owner of a slot may use it to send information. The ownership of static slots is fixed, so it is always the same node that owns a given static slot, and all nodes on the bus know who owns a static slot. The ownership of dynamic slots varies, and is negotiated by the nodes through the static slots.

The basic principles of the DTM protocol are as follows. Suppose a host (the sender) wants to transfer data to another host (the receiver). The sender tells this to its node, and what transfer capacity it needs. The node allocates dynamic slots to the sender host, and sends in its static slot a control message addressed to the receiver. This control message tells the receiver's node what slots have been allocated to the transfer. The receiver's node will then, for each cycle, take the content of these slots and pass it to the receiver. When the transfer is completed, the sender's node sends another control message to the receiver's node, which then stops passing slots to the receiver.

The sender's node must own the dynamic slots it allocates to the sender. If it does not have dynamic slots enough to give the requested capacity, it must try to get more dynamic slots. It does this by sending in a static slot a request for dynamic slots. A node that receives this request, and has unused dynamic slots, may give some of its unused slots to the sender's node (by sending a control message in a static slot).

2.3 Channels

The DTM network provides a service based on *channels*. A channel is a portion of bandwidth (i.e., a set of slots) with a sender and an arbitrary number of receiver hosts. A sender sends data on the channel to the receivers; it is guaranteed that the data will reach the receivers at the rate given by the bandwidth of the channel. Channels are synchronous, and can be seen as continuous streams of data from senders to receivers. Data sent on channels cannot be lost, since DTM is a network free of contention, but data can of course be corrupted which, as explained earlier, is not detected or corrected in DTM. The service a node offers its hosts is based on the following four operations.

- The *create* operation instructs a node to allocate a given amount of slots to a new channel. The node may fail to get slots for the channel, in which case the channel cannot be created. We say that the node that creates a channel is the *controller* for that channel.
- The *remove* operation instructs a node to deallocate slots from a channel. A host must have *created* a channel to be able to *remove* it.
- A host can request its node to *attach* it to a channel, as sender or receiver. A host attached as receiver will be provided the data in the slots of the channel. If a host is attached as sender, the node moves data from the host into the slots of the channel. DTM does not guarantee that there is only one sender at a time on a channel, this is left to the application. As a measure of security, a host that creates a channel must indicate whether or not other hosts may attach as senders. If not, only the host that created the channel may send on it.
- The *detach* operation tells a node that the host no longer wishes to send/receive data on a channel.

2.4 DTM Protocol

Data can be sent on channels only in one direction, from sender to receivers. However, channel control information can be sent also from receiver to sender, so channel control is bi-directional. The DTM protocol is based on a set of *control messages* sent in static slots. A node can send a control message upon request from its host, in response to control messages from other nodes, and spontaneously for management purposes. A control message is represented as a 64-bit word, so it fits in one slot, and consists of a type field, a destination address and arguments. The source address is implicit—only the owner of a static slot may put data in it, and all nodes know who owns a static slot.

There are two types of addresses in DTM; *host* addresses, identifying individual hosts, and *multicast* addresses, identifying logical groups of hosts. There are three categories of DTM control messages; for channel-slot mapping, sender/receiver synchronization, and slot reallocation.

2.4.1 Channel-Slot Mapping

- The `announce` message announces that a channel exists, and what slots are associated with the channel. When a host requests a channel to be created, its node may announce this to other nodes. `Announce` messages are also sent in response to `query` messages, see below.
- A node may ask by a `query` message whether a given channel exists, and what slots are associated with it. If the channel exists, its controller node responds to the `query` with an `announce` message. The intention is that nodes should maintain a cache of `announce` messages they have received on the fibers. When a host requests attachment to a channel, the node consults its cache to see what slots corresponds to the channel. If a channel is not in the cache, the node may ask by sending a `query` message that the `announce` message for the channel should be repeated. This scheme is similar to the Internet Address Resolution Protocol (ARP) for translating Internet addresses to Ethernet addresses [16].
- A `remove` message indicates that a channel has been removed. A node that has hosts attached to a channel for which it receives a `remove` message, signals to the hosts that the channel has disappeared, and stops sending/receiving data on the channel.

2.4.2 Sender/Receiver Synchronization

When a hosts attaches/detaches a channel, the node can signal the channel's controller by sending an `attach` (`detach`) control message. The controller will propagate the message to its host, but the message causes no further actions in the controller node. Thus, higher-layer protocols decide how and when these message are used. For instance, they can be used for sender/receiver synchronization before and after data transfer, and for accounting purposes.

2.4.3 Slot Reallocation

A node that needs more dynamic slots can request more slots using a three-way handshake scheme. First the node sends a `slot transfer request` message. A node that receives such a message, and has unused dynamic slots, may give slots away. It does this by responding with a `slot transfer` message. The node that requested slots confirms that it has received more slots by sending a `slot transfer confirm`. It could happen that several nodes give slots away, and the node gets more slots than it needs. The node may then reject some of the offered slots, for this the node uses a `slot transfer reject` message.

A node that gives slots away remains the owner of those slots until it receives a `slot transfer confirm` or a `slot transfer reject`. In this way a slot can never be without an owner.

Decision about when and which node to ask for slots are made according to a *slot reallocation algorithm*[19]. The goal of the reallocation algorithm is to efficiently move bandwidth to where it is currently needed.

2.5 Service

The DTM operations can be combined into many different kinds of services. We next describe what we think will be the common ways of using them.

2.5.1 Point-to-point Connections

A *point-to-point connection* is a channel with one sender host and one receiver host that synchronize before and after the data transfer on the channel. Point-to-point connections can, for instance, be used for telephone calls and file transfers. The procedure for setting up point-to-point connections in a DTM network are as follows:

The sender creates a channel and asks its node to announce the channel to the receiver with an `announce` message to the receiver's host address. The receiver is waiting for the channel to be announced, and when it receives the announcement it responds with an `attach`, thereby acknowledging the connection. The sender or the receiver may close the connection by sending a `remove` or a `detach`, respectively.

2.5.2 Broadcasts

A *broadcast* is a channel with an arbitrary number of receivers, where the sender need not synchronize with the receivers. Such channels are typically used for broadcasting audio and video. Broadcast in datagram networks is commonly implemented as sending datagrams with a "wild card" destination address that every host on the network recognizes. DTM broadcasting is similar, in the sense that it is only the destination address that distinguishes broadcast communication from two-party communication. In DTM, however, continuous streams of data are broadcast instead of individual datagrams.

The sender starts by creating the channel and announcing it with an `announce` on a multicast address. Receiver hosts can then quietly “tune in” to the channel by requesting to be attached as receivers to the channel, without announcing this to the controller of the channel. Correspondingly, a host that wants to disconnect a broadcast channel simply asks its node to detach the channel.

2.5.3 Selective Multicasts

The *selective multicast* is a hybrid between the two types of services previously mentioned. It is a channel with several receivers where the sender synchronizes with the receivers before and after transfers. It can be regarded as several point-to-point connections, all with the same sender and using the same slots.

The sender creates the channel and announces to all receivers, one by one using their host addresses, that the channel exists and what slot it uses. Receivers acknowledge with `attach` messages that they have started receiving, and also announce through `detach` messages when they disconnect from the channel.

2.5.4 Segmentation and Reassembly

A DTM channel is a stream of slots that arrive at regular intervals, so a channel provides synchronous transfer of small units of data. Some applications, in particular in the area of traditional computer communication, prefer asynchronous transfer of large packets. The purpose of the DTM Segmentation and Reassembly (SAR) protocol is to provide such a service. The DTM SAR protocol is based on an *idle* pattern, which is a special 64-bit word sent on a channel between packets, that is, while the channel’s sender has nothing to send. When the sender requests a packet to be sent, the node precedes the packet with a 64-bit *packet start* word indicating the length of the packet. The receiver node detects the packet start on the channel (the packet start cannot be confused with the idle pattern), and arranges for the corresponding amount of subsequent 64-bit words to be received by the host. When the packet has been transferred, the sender node goes back to sending *idle* patterns. The service provided by this protocol is similar to that of SAR protocols in other small cell networks, such as ATM and DQDB[8][9].

The packet start word has an *alert* flag. This flag tells the receiver node to alert its host (by an interrupt, for example) when the packet has been received. The usage of this flag is discussed further in section 4.4 on page 16.

3.0 The DTM Network

Hosts in a DTM network are connected to nodes which are connected via the medium. Any node connected to two or more dual buses can switch data between these, so switching is decentralized in DTM networks. DTM switching is synchronous, which has the advantages that contention is avoided and that a node needs no extra buffering capacity to be able to switch. The drawback is that buses in a DTM network must be synchronized. This is done on a cycle basis—cycles have the same duration on all buses.

Several dual buses can be interconnected into a mesh structure with a few switching nodes (Fig. 4). The switching capacity of a network is increased by increasing its amount of switching nodes..

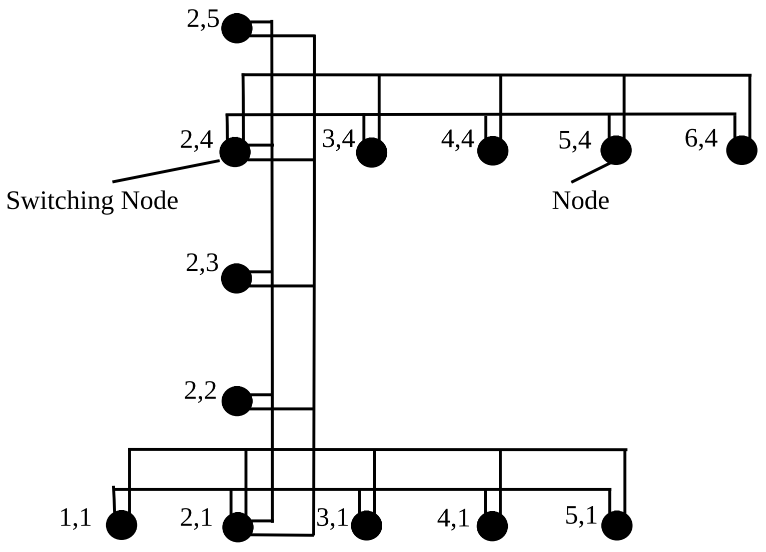

FIGURE 4. DTM mesh topology.

In order for hosts on different dual buses to communicate, channels on the fibers from the sender to the receiver have to be linked together into a *route*. Each switching node on the route is responsible for creating an outgoing channel on its bus. For example, if node (5,1) wants to communicate with node (5,4) in Fig. 4, it first creates a channel to (2,1). Node (2,1) creates a channel to the next switching node, here (2,4), which finally creates a channel to (5,4). When a node has created the outgoing channel, the node starts switching—copying slots from incoming to outgoing channel. This means that when the route is established, switching requires no processing of the data stream. Provided that a switching node can buffer one cycle of data for each of its buses, there can be no contention or overflow in the node.

An attempt to set up a route fails if a switching node on the route does not have enough free slots to create a channel. Then another route has to be tried. A routing algorithm, which can set up a channel by sending a single static slot from sender to receiver, is implemented in the prototype. However, this algorithm imposes limitations on the number of hops on a route, and thus on the network topology, and we will develop a more general routing algorithm.

3.1 Synchronization

Different dual buses in a DTM network can run at different speeds. This makes it possible to update and increase the speed of parts of the network without forcing all nodes to be updated. Since the cycle time and the slot length is constant throughout the DTM network, buses running at different bit rates can be connected to the same node (Fig. 5) and switching can be done between these buses.

DTM uses an asynchronous scheme for synchronization of cycles on different buses. Each cycle starts with a *start* slot and is followed by one or more *fill* slots The fill slots are for filling out gaps between cycles (cycles have fixed length, so there is no need to mark the end of a cycle). To get the same cycle length on all fibers in the network, the beginning of cycles on different fibers are synchronized. This is done with a hierarchical synchronization scheme with two types of special nodes: one *master* node and many *slave* nodes. These special nodes *control* some of their outgoing fibers, which means that they are responsible for generating cycles and also clock signals on these fibers. The master controls all its outgoing fibers, and uses a local clock to generate cycles periodically, by sending start slots. A slave node has a *trigger* fiber, which is one of its incoming fibers. When a slave receives a start slot on its trigger fiber, it sends start slots on the fibers it controls . After each cycle, the slave sends fill slots until it receives a new start slot on the trigger fiber. There will be a propagation delay from the start of the master's cycle until the slave starts its cycle, but this delay is constant and does not interfere with the synchronization (Fig. 5).

.

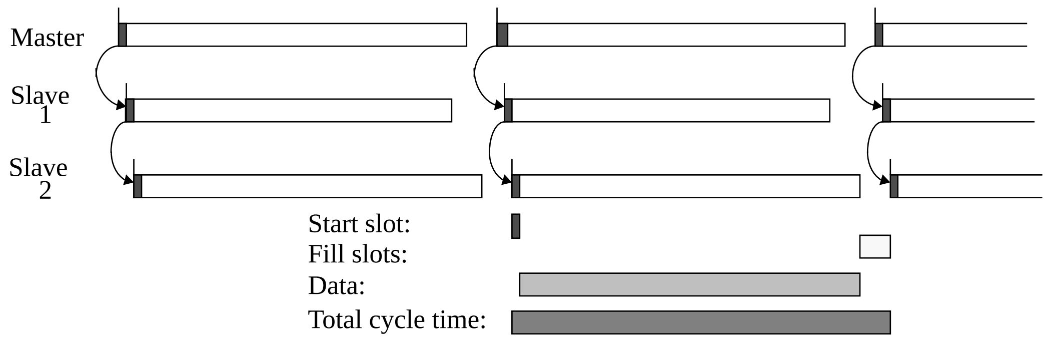

FIGURE 5. Cycle start of different nodes

Each master and slave has a local clock for bit-level synchronization. They encode the clock signal into the data, and the other nodes extract the clock from the incoming data through a phase-locked loop.

The master resides at the beginning of all its buses and there is one slave at each end of a dual bus, controlling the fiber going out from that end. One of the slaves has as its trigger fiber the incoming fiber of the dual bus, and the other slave an incoming fiber from another dual bus. One of the slaves therefore must be attached to more than one dual bus. For instance, in Fig. 4 node (1,1) could be master and control its outgoing fiber. Node (5,1) controls the other fiber of the same dual bus and is triggered by the fiber controlled by the master. On the vertical bus, nodes (2,1) and (2,5) are slaves and (2,1) is triggered by the fiber controlled by the master. On the upper dual bus, (2,4) and (6,4) act as slaves. Node (2,4) is thus triggered by the start slot from (2,1) on the vertical bus and (6,4) is triggered by the fiber controlled by (2,4).

This scheme restricts the possible topologies for DTM networks since one end of a dual bus must be connected to another dual bus. It appears that this restriction can be removed by introducing a third type of synchronization node, but this is a subject for further study.

4 System prototype

The main goals in the design of the prototype were that it should be flexible and simple. Therefore the prototype node is mostly in discrete components, and interfaces directly to optical fibers.

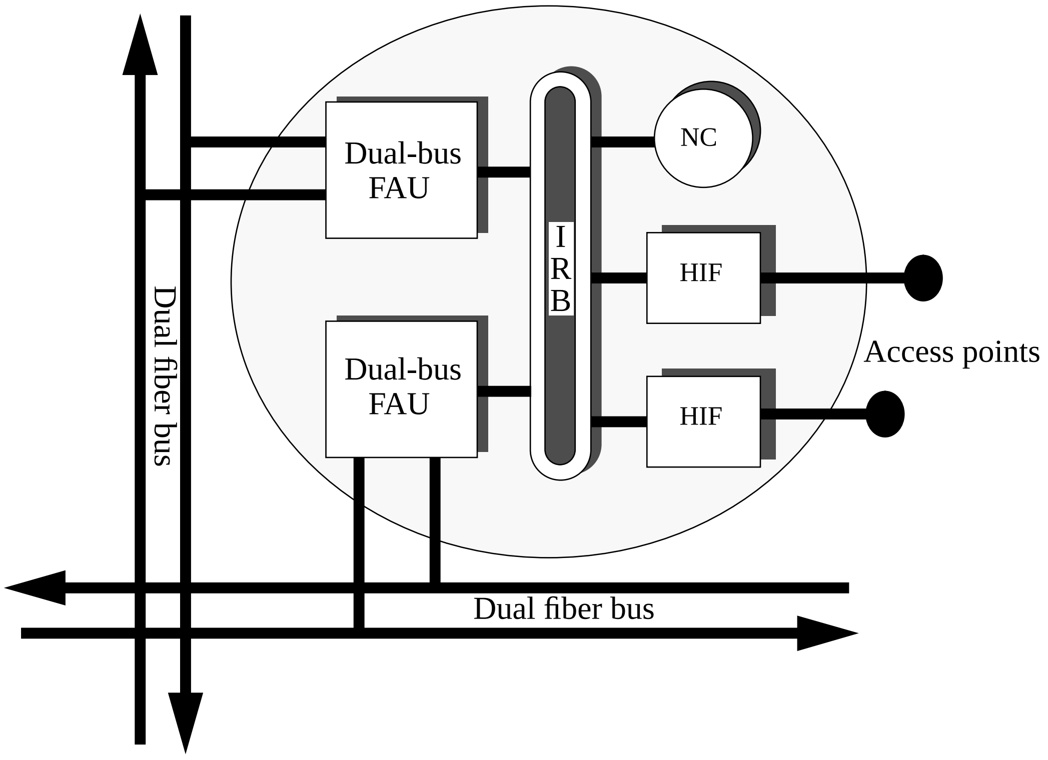

FIGURE 6. Prototype node

A prototype node consists of three types of units. First, a node has one or two Fiber Access Units (FAU), each connecting the units in the node to one dual bus. Second, a host gets access to the DTM services through a Host Interface (HIF). There can be up to two HIFs in a node in the prototype. Third, a Node Controller (NC) controls the node and implements the DTM protocol. The units in a node are connected by a 76 bit wide Internal Ring Bus (IRB), see Fig. 6.

4.1 FAU

The FAU transfers data between the fiber and the other units in the node by multiplexing/demultiplexing between channels and slots. Its operation is table-driven; it has one *channel table* for outgoing data and one for incoming, and each table has one entry per slot in a cycle (see Fig. 7). The channel table tells the node whether a slot is *open* in the node, meaning that the slot is allocated to a channel, and that there is a unit in the node attached to that channel (the Open/Close field in Fig. 7). If the slot is open, the entry in the table identifies the channel.

The FAU uses a *slot counter* as index into its tables. The slot counter contains the *slot number* (i.e., the position in the cycle) of the current slot on the fiber. For each slot on the fiber, the FAU first consults the channel table for incoming data. If the slot is open in that table, the FAU takes the content of the slot and sends it on the IRB together with its channel identifier. The unit attached as receiver to a channel looks for data with that channel's identifier on the IRB, so the channel identifier is an implicit unit address.

The FAU thereafter consults the channel table for outgoing data. Since transfer time over the IRB depends on the distance between sender and receiver unit, the FAU buffers one cycle of outgoing data in the *cycle buffer.* The cycle buffer is a table indexed by slot number, just as channel tables, and each entry is a one-place slot buffer. If a slot is open in the outgoing table, its cycle buffer entry contains the data to write into the slot. After writing this data in the slot, the FAU has to refill the cycle buffer entry. The FAU does this by sending on the IRB a request for more data. This request contains the channel's identifier; a unit attached as sender to a channel looks for data requests with that channel identifier on the IRB. When a unit receives such a data request, it sends more data to the FAU over the IRB.

The FAU behaves in the same way for all slots and for all units in the node: If a dynamic slot belongs to a channel to which a host is attached, the FAU transfers data between the fiber and a HIF. If a dynamic slot belongs to a channel that the node switches, the FAU transfers data between the fiber and another FAU. Static slots are all regarded as belonging to a special channel to which the Node Controller is attached, so static slots are transferred between fiber and NC.

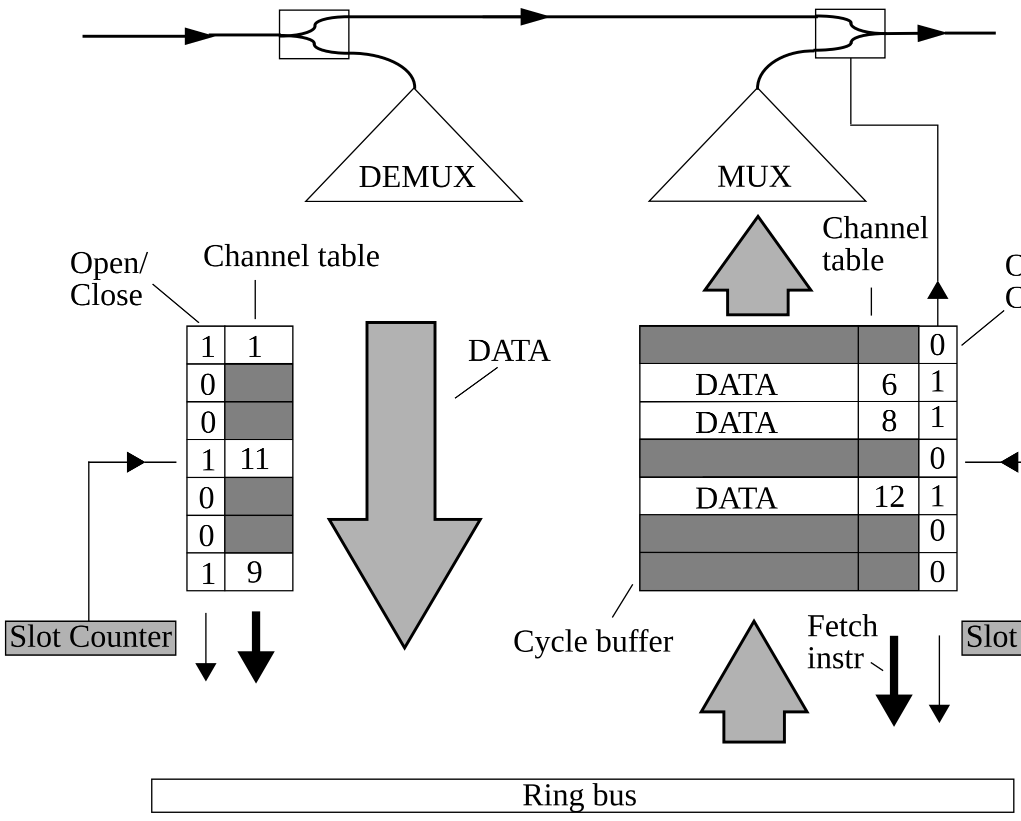

FIGURE 7. Fiber Access Unit (FAU)

Most of the logic in the FAU operates on slots. This implies that large slots yield low-speed FAU electronics. However, large slots give wide data paths, more components and less granularity of bandwidth fragments. We consider 64-bit slots to be a good compromise. At 622 Mbit/s fiber rate, channel tables and other control logic will operate at approximately 10 MHz, and the smallest possible fragment of bandwidth will be 512 kbit/s. At 5 Gbit/s, the operating speed would be less than 80 MHz.

4.2 Internal Ring Bus (IRB)

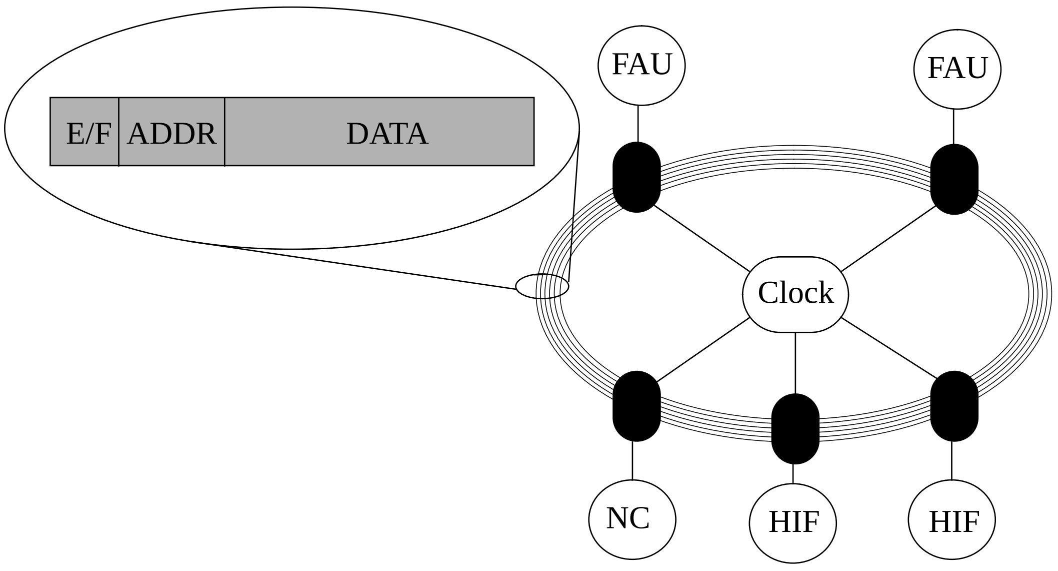

FIGURE 8. The Internal Ring Bus

The IRB is a *slotted ring bus*, which is a simple high capacity bus. There is one IRB slot for each pair of adjacent units on the IRB (see Fig. 8). The IRB is a parallel bus with 76 bits of data, an E/F flag, and a four bit address. The 76 bits of data are for storing one 64-bit data word and a 12-bit channel identifier. The E/F flag indicates whether the IRB slot contains valid data. The four bit address indicates the destination unit of the IRB slot. The IRB is a synchronous bus, where IRB slots are “clocked” around the ring one hop at a time. The destination unit clears the E/F flag when it receives the IRB slot.

To prevent contention, the IRB is over-dimensioned. The bus is clocked at 40 Mhz which yields a capacity of 2.56 Gbit/s. This is more than necessary; it guarantees each node an IRB access delay shorter than the time it takes to send a slot on a fiber.

4.3 Node controller

The Node Controller (NC) has two functions. First, it controls the other units in the node, and communicates with them over the IRB. For instance, the NC sets up the channel tables in the FAU by writing data to it over the IRB.

Second, the NC implements the DTM protocol. This includes mechanisms for maintaining channels, and algorithms for slot reallocation and routing. The NC communicates with other nodes through static slots.

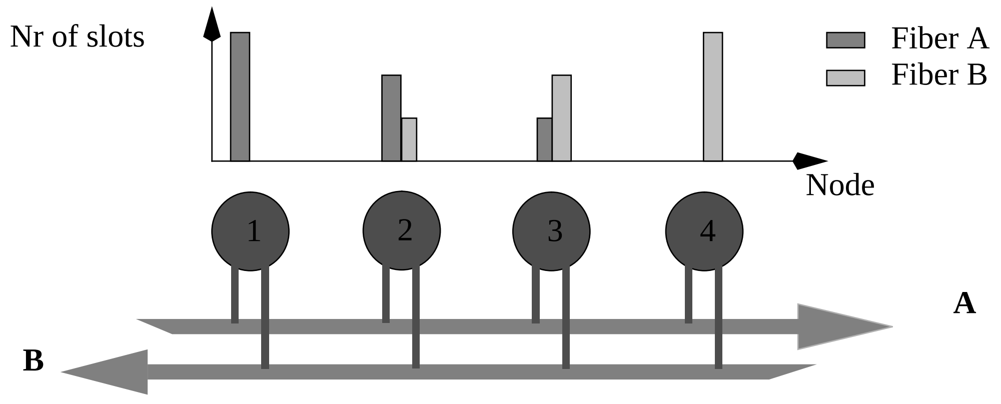

FIGURE 9. Initial slot distribution

The slot allocation algorithm in the prototype uses an asymmetric initial distribution of slots. Initially, the node at the beginning of a fiber gets the most slots. The allocation then decreases linearly so the last node on the fiber is without slots (see Fig. 9). A node periodically tells other nodes how many free slots it has by sending a status message. A node that needs more slots requests this from the node with most unused slots. A node is considered to need more slots when its number of free slots has decreased to a certain fraction of the initial value. Further, a node does not give away all slots. It keeps some slots, so that it can immediately satisfy a request for a channel with little bandwidth.

For flexibility, we use a Sun SPARCengine 2 as node controller. In future implementations we will use an embedded processor. Nodes that do not switch may implement the NC in the host.

4.4 Host Interface

The prototype has interfaces for Sun SPARCstation hosts. These HIFs implement the SAR protocol, which converts the stream of slots into larger variable-length packets, and vice versa. There are two major goals in the HIF design. First, it is costly to interrupt the CPU, in particular if it is a RISC processor, so the HIF should generate few interrupts to its host. Second, copying of data between primary memory and host interface takes a lot of time. Thus, the design of the HIF should not force DTM drivers to copy data between host memory and the HIF. Similar design considerations have been made for other host interfaces, see [3][21].

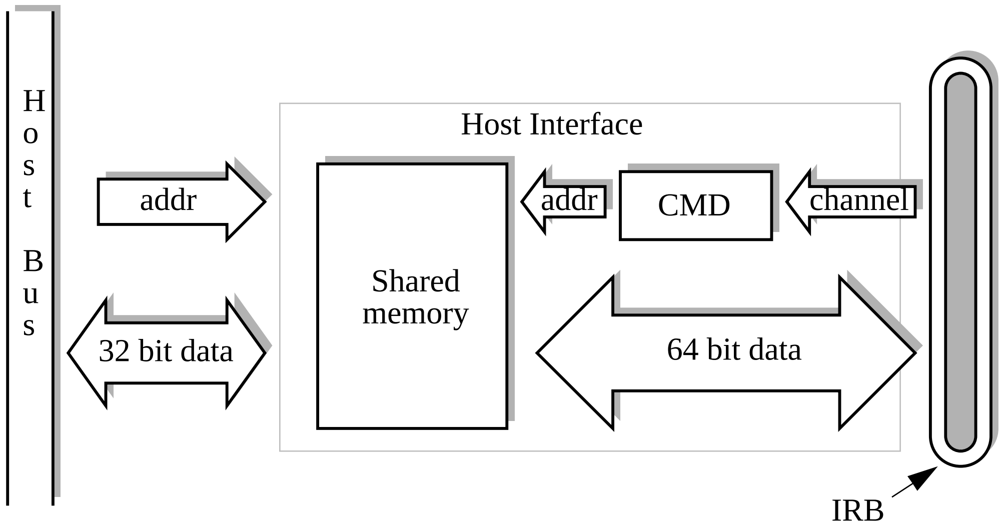

FIGURE 10. Host interface

To avoid copying of data the HIF is based on a shared memory accessed both by the host and the HIF. Such a design has been used also by others for the same reason, see for instance [3]. In the shared memory, the host writes data to be sent on the network, and reads data received from the network. Control information to/from the host is read/written in a special area in the shared memory.

The shared memory is divided into equally sized buffers (of typically 1–10 Kilobytes), where each buffer can store one packet. We chose this buffer scheme since it is easy to implement in hardware. The drawback with the scheme is that if packets are much smaller than buffers, then the scheme wastes memory. Alternative solutions are, for instance, variable-sized buffers, and to use small buffers and allow packets to span over several buffers.

There is a queue of buffers from the HIF to the host. In this queue, the HIF puts buffers with packets received from the network, along with empty buffers (buffer whose contents have been sent to the network). The HIF can notify the host by an interrupt that a buffer has been put into the queue. However, we do not want the HIF to generate an interrupt each time it puts a buffer in the queue. Instead we let the host that *sends* on a channel decide, for each packet, whether interrupts should be generated. The sender uses the alert flag in the SAR protocol for this. We motivate this design by claiming that it reduces interrupts. A busy-wait scheme, such as polling, occupies the CPU and is therefore not practical in a time-sharing environment. A common interface design is to unconditionally generate an interrupt when a packet is sent/received. Such schemes can generate unnecessary interrupts. For example, it is probably sufficient to generate a single interrupt for two packets that arrive back-to-back. We have therefore introduced conditional interrupts, which makes it possible to generate less than one interrupt per packet.

The Channel Multiplexer/Demultiplexer (CMD) controls the flow of data between the node and the shared memory (Fig. 10). The CMD is implemented as a table with one entry for each channel. Each entry consists of the address of the currently used buffer, its *working point* which is an offset into the buffer indicating where the next 64-bit word should be read/written, and a channel state variable (a channel has three different states, *idle*, *receiving* and *receiving with interrupt*).

4.5 The testbed

The first prototype network (Fig. 11) covers a distance of approximately 30 km. Swedish Telecom has provided the MultiG programme with "dark fiber", optical fiber for exclusive use within the project. This fiber connects three sites in Stockholm and its vicinity:

• SICS (Swedish Institute of Computer Science) in Kista (a suburb of Stockholm),
• CClab at KTH (Royal Institute of Technology) in Kista,
• NADA at KTH in Stockholm.

The optical fibers are single mode fibers with low attenuation which allow them to be used without repeaters. The three sites will be connected by two dual buses, which makes it possible to experiment with switching between dual buses.

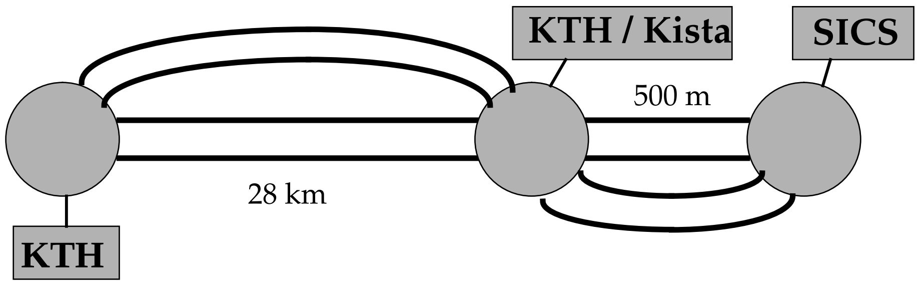

FIGURE 11. The first prototype testbed

As a next step, DTM will be tested in a larger test environment in the Stockholm Gigabit Network (SGN) [11]. SGN is a joint project between research organizations and industry in and around Stockholm to build a network for gigabit/sec rates. The network will be used for research on high-speed MAN transmission, control and applications. For a survey of other gigabit testbeds, see [6].

5 Related Work

DTM provides a service based on connections with resource reservation, similar services are provided by plaNET's Packet Transfer Mode (PTM) [2] and ATM [9]. This makes it possible to guarantee that data units arrive at their destination within time limits. In DQDB, on the other hand, there is no resource allocation for channels. Thus, DQDB can only guarantee that a slot eventually reaches its destination.

The broadband switch SpectraNet [14] by AT&T is for synchronous switching of SONET/SDH traffic[1]. DTM switching nodes use a similar scheme, whereas Autonet [18] and ATM-based networks use asynchronous switching. The advantage of synchronous switching is that there can be no overflow or contention in switches. A synchronous switch must, however, coordinate its incoming and outgoing links. Most switching networks use special-purpose switching nodes, see for instance [5] [12]. We have chosen to use general-purpose nodes in DTM, so the same kind of nodes are used for switching and for connecting hosts to the network.

Networks such as Hangman [22], QPSX MAN [13], ATM networks and telephone networks [20], use fixed sized cells. The alternative is to use variable-sized packets, as in PTM and Autonet. Fixed sized cells simplify network interface hardware, but require more processing in higher-layer protocols (SAR). For DTM, we chose fixed sized slots.

6 Conclusions and Further Work

DTM is a synchronous TDM-based fiber-optical network, that features support for dynamic reallocation of bandwidth. It provides a service based on multicast channels, and network resources are reserved on a per channel basis. The main advantages with this scheme is that channels provide services with well-defined real-time properties, and that there can be no loss of data due to network contention.

We have described and motivated the medium access technique and the protocol that provide the DTM service. Further, we have shown how dual buses are combined into larger networks, and explained how data is (synchronously) switched between dual buses. Finally, we described the DTM testbed being developed. This testbed is both a prototype implementation of DTM, and a platform for experiments with distributed multimedia applications. It runs at 622 Mbit/s full duplex and connects three sites spread out over a distance of 30 km.

The dynamic reallocation of bandwidth is a major feature in DTM, and we are currently investigating several algorithms for slot reallocation. These algorithms do not lend themselves easily to analytical studies, so we use simulations as the main method of evaluation.

A natural extension to slot reallocation appears to be *slot reuse*[17] [10]. This increases the capacity of the network above the nominal bit rate of the fiber, by allowing several nodes to use the same slot on different intervals on the bus. The basic modification is that nodes no longer own slots globally on the bus. Instead, they own slots on intervals on the bus. To create a channel, the node must own slots on the entire interval between sender and receiver.

We plan to extend the DTM protocol to use multiple optical wavelengths, as for example in the Rainbow networks [7]. We believe DTM is well-suited for multiple wavelength networks, since it does not use read-modify-write operations, and has been designed for all-optical networks. One of the advantages of this is that FAU electronics only has to be designed for the speed of a fraction of the wavelengths.

REFERENCES

[1] R. Ballart and Y. C. Ching, “SONET: Now It’s the Standard Optical Network”, IEEE Commun. Mag, Vol. 29, No. 3, March 1989.

[2] I. Cidon, I. Gopal and R. Guérin, “Bandwidth Management and Congestion Control in plaNET”, IEEE Communication Magazine, Vol. 29, Oct. 1991.

[3] B. S Davie: “A Host-Network Interface Architecture for ATM” In Proc. ACM SIGCOMM ‘91, Zurich, Sep. 1991.

[4] L. Gauffin, L. Håkansson and B. Pehrson, “Multi-gigabit networking based on DTM,” Computer Networks and ISDN Systems, Vol. 24, No. 2, 1992.

[5] J.N Giacopelli et al., “Sunshine: A High Performance Self-Routing Broadband Packet Switch Architecture”, IEEE J. Sel. Areas Commun., Vol. 9, No. 8, pp 1289-98, Oct 1991.

[6] “Gigabit Network Testbeds”, IEEE Computer, Vol. 29, No. 9, Sept 1990.

[7] P.E. Green, “An All-Optical Computer Network: Lessons learned”, IEEE Network, Mars 1992.

[8] M. Hill, A. Cantoni, T. Moors, “Design and Implementation of an ATM Receiver” , 3rd International IFIP WG6.1/6.4 Workshop on Protocols for High-Speed Networks, Stockholm, May 13-15, 1992.

[9] R. Händel and M. N. Huber, “Integrated Broadband Networks,” Addison-Wesley, 1991.

[10] A. E. Kamal, “ Efficient Multi-Segment Message Transmission with Slot Reuse on DQDB” Proc. INFOCOM ‘91, pp 869-878.

[11] G. Karlsson, “Stockholm Gigabit Network: An Experimental Broadband Network for Transmission, Networking and Applications Research”, Proc. of the ERCIM Workshop on Network Management, Heraklion, Crete, Greece, October 29-30, 1992, pp. 1-9.

[12] M. J. Karol et al. AT&T Bell Laboratories, “Hierarchical Gigabit ATM Switching: Applications and Performance”, Proc. of XIV Int. Switching Symposium 1992, Yokohama

[13] R. M. Newman, Z. L. Budrikis and J. L. Hullett, “The QPSX Man,” IEEE Communications Magazine, Vol. 26, No. 4, April 1988.

[14] W. Payne, M. F. Parker, R. L. Pawelski, J. Montsma, “SpectraNet: A High Speed, Synchronous Switch for Broadband STM Communication”, Proc. of XIV Int. Switching Symposium 1992, Yokohama

[15] B. Pehrson, S.Pink and P. Gunningberg, “MultiG—a research program on distributed multimedia applications and gigabit networks”, IEEE Network Magazine, Vol. 6, No. 1, January 1992.

[16] D. Plummer, “An Ethernet resolution protocol,” RFC 826, SRI Network Information Center, Menlo Park, CA, November 1982

[17] Oran Sharon and Adrian Segall, “A simple scheme for slot reuse without latency in a Dual Bus”, 3rd International IFIP WG6.1/6.4 Workshop on Protocols for High-Speed Networks, Stockholm, May 13-15, 1992.

[18] M. D. Schroeder et. al., “Autonet: a High-speed, Self-configuring Local Area Network Using Point-to-Point Links”, Digital Equipment Corporation, SRC Research Report 59, Apr. 1990.

[19] G. Sjödin, “An algorithm for handling slot allocation in DTM”, Proc. of the 4th MultiG Workshop, Stockholm, May 12,1992.

[20] W. Stallings, “Data and computer communications”, Macmillan Publishing Company, 1991.

[21] C.B.S. Traw and J.M. Smith, “A High-Performance Host Interface for ATM Networks”, In Proc. ACM SIGCOMM ‘91, Zurich, Sep. 1991.

[22] G. Watson, S. Ooi, D. Skellern and D. Cunningham, “Hangman Gb/s Network”, IEEE Network, July 1992.