WDM Burst Switching for Petabit Capacity Routers

Yuhua Chen and Jonathan S. Turner
Washington University, St. Louis
{yuhua,jst}@cs.wustl.edu

Abstract

WDM burst switching is an approach to building very high capacity routing switches based on optical data paths and electronic control. Burst switches assign user data bursts to channels in WDM links on-the-fly in order to provide efficient statistical multiplexing of high rate data channels. The overall system architecture is designed to facilitate the introduction of optical switching components as they become more highly integrated. At the same time it exploits the sophistication of modern electronic processing to perform the routing and higher level control operations needed in realistic, large scale networks.

1 Introduction

Bandwidth requirements for both civilian and military applications are growing exponentially and data network capacities are now exceeding voice network capacities. Satellites and other data gathering systems are producing massive amounts of information that must be efficiently transported to remote locations for storage and analysis. The emergence of WDM technology is making it possible to move these massive data sets through fiber optic data links, leading to lower costs, further fueling demand. While ATM switches and IP routers can switch data using the individual channels within a WDM link, this approach implies that tens or hundreds of switch/router interfaces must be used to terminate a single fiber. Moreover, there can be a significant loss of statistical multiplexing efficiency when WDM channels are used simply as a collection of independent links, rather than as a shared resource. To meet the growing demands of military and civilian applications in the next century, it will be necessary to build routing switches with aggregate capacities exceeding a petabit per second. It appears likely that petabit routing switches will have to use optical switching technology to move the data between input and output links. Unfortunately, previous efforts to design effective optical packet switches have been disappointing, limited both by the primitive stage of development of optical processing and by the lack of a comprehensive vision of the system as a whole.

[0]This work is supported by the Advanced Research Projects Agency and Rome Laboratory (contract F30602-97-1-0273).

WDM burst switching [7, 8] seeks to overcome the limitations of previous optical packet switches by more effectively exploiting the complementary strengths of optical and electronic technologies. In a burst switched network, each link comprises a number of WDM data channels and one (or possibly more than one) control channel. The control channel(s) carries *Burst Header Cells* (BHC) that describe bursts carried within the WDM data channels. This separation of data and burst-level control information allows the switches to propagate the data without ever converting it to electronic form, while still allowing the control information to be processed electronically. The control subsystem of the burst switch determines the output port that an arriving burst is to be forwarded to (using an IP address embedded in the BHC), selects a free channel within that outgoing link and a path through the switch to reach the required output link and channel. Bursts are variable in length and may be as short as hundreds of bytes or as long as millions of bytes. In order for burst switching systems to be competitive with electronic routers, they must be able to handle short bursts efficiently, as well as long bursts.

An earlier paper [8], provides a more complete exposition of the basic concepts of burst switching and explains the principal performance concerns that drive the design of large scale burst switching systems. In this paper, we describe a burst switch architecture that can be scaled to support thousands of links, with potentially hundreds of WDM channels per link. At channel rates of 10 Gb/s, this translates to terabit capacity links and petabit capacity routing switches. We describe a candidate design for the key optical data path component in this architecture and quantify the number of optical components required for its implementation. This allows calculation of component cost objectives that will have to be met in order for optical technology to displace electronics in the data paths of future, high capacity routers.

The control of burst switches requires the development of new mechanisms for managing the channels in WDM links. To enable efficient management of short data bursts, as well as long data bursts, the control system must make

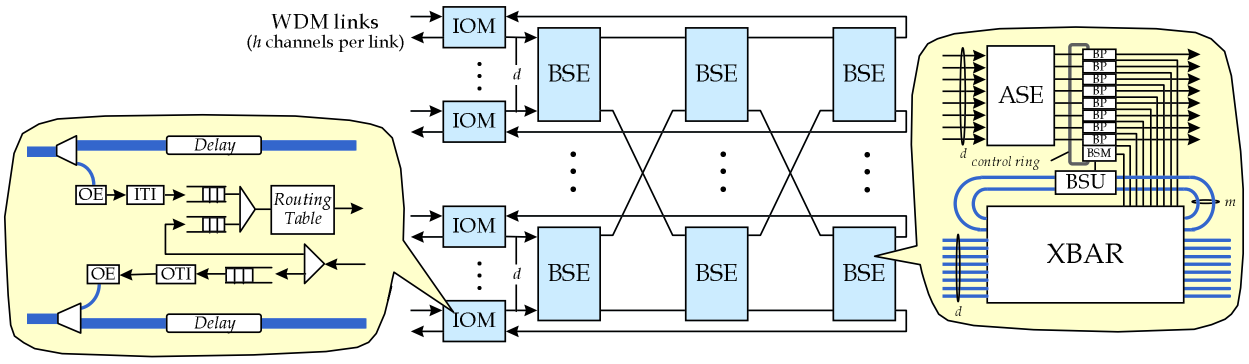


Figure 1: Burst Switch Architecture

resource management decisions based on projected future resource availability. The resulting *look-ahead resource management* mechanisms raise new challenges and opportunities for switch designers.

2 Overall System Architecture

Figure1 illustrates a scalable burst switch architecture consisting of a set of *Input/Output Modules* (IOM) that interface to external links and a multistage interconnection network of *Burst Switch Elements* (BSE). The interconnection network uses a Beneš topology, which provides multiple parallel paths between input and output ports. A three stage configuration comprising d port switch elements can support up to d^2 external links (each carrying h WDM channels). The topology can be extended to 5,7 or more stages. In general, a $2k-1$ stage configuration can support up to d^k links, so for example, a 5 stage network constructed from 8 port BSEs supports $8^3 = 512$ links. If each link carries 512 WDM channels at 10 Gb/s each, the aggregate system capacity exceeds 2.5 petabits per second.

As shown in Figure 1, each IOM contains a control section which processes the Burst Header Cells (BHC) received on the control channel. The address information in the BHC is used to select an entry from the routing table, containing the output port number that the burst associated with that BHC is to be forwarded to. This output port number is added to the BHC as it is passed to the first BSE.

When a BHC is passed to a BSE, the control section of the BSE uses the output port number in the BHC to determine which of its output links to use when forwarding the burst. If the required output link has an idle channel available, the burst is switched directly through to that output link. If no channel is available, the burst can be stored within a shared Burst Storage Unit (BSU) within the BSE. In the first stage of a three stage network, bursts can be routed to any one of a BSE's output ports. The port selection is done dynamically on a burst-by-burst basis to balance the traffic load throughout the interconnection network. In general, the first $k-1$ stages of a $2k-1$ stage network perform dynamic traffic distribution to balance the load as evenly as possible. This yields optimal scaling characteristics, making it possible to build large systems in which the cost per port does not increase rapidly with the number of ports in the system.

3 Cost Considerations

The complexity (and cost) of this system is determined primarily by the crossbars within the BSEs. A candidate design for an all optical crossbar is shown in Figure 2. The crossbar is divided into separate sections, one for each output. Within each of these sections, we have a $d \times h$ crossbar, h *wavelength selectors* and h *wavelength converters*. These are followed by a passive optical coupler which combines the signals back onto a single fiber. Each wavelength selector allows a specified input wavelength to propagate to its output port. Each wavelength converter uses its input optical signal to modulate an optical carrier at a fixed output wavelength. Thus, a $d \times d$ BSE with h channels per link requires d^2h optical crosspoints, dh wavelength selectors and dh wavelength converters. A system with n external links requires $(n/d)(2k-1)$ BSEs where $k = \log_d n$. Thus, for each external channel, we require $(2k-1)d$ optical crosspoints plus $(2k-1)$ wavelength selectors and wavelength converters. So, for $d = 8$ and $k = 5$ we require 72 optical crosspoints per external channel and 9 wavelength selectors and converters.

This analysis can be used to get some insight into the potential for optical technology to replace electronics in the data paths of future routing switches. It is now technically feasible to build the switching fabric for terabit capacity routing switches using inexpensive CMOS integrated circuits. The best architectures require less than one switching chip per Gb/s of system capacity [1], even for large configurations. Larger systems require optical inter-

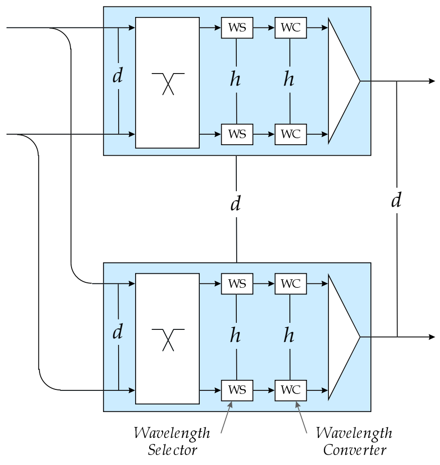


Figure 2: Crossbar

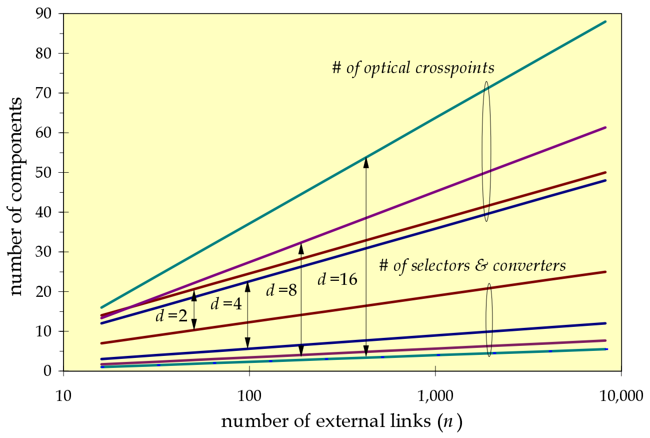


Figure 3: Number of Selected Data Path Components per External Channel

connection between different equipment racks, but newly available optical interconnect components based on VCSEL arrays make such interconnects reasonably economical [5]. This allows interconnection networks for large electronic routing switches to be constructed at a parts cost in the neighborhood $300 to $500 per Gb/s of capacity.

Thus, the core interconnection network for a WDM switch supporting channels operating at 10 Gb/s should have a parts cost of no more than about $3,000 to $5,000 if it is to be competitive with current generation electronics. So the example system discussed above with $d = 8$ and $k = 5$ will become economically viable (relative to current generation electronics) when $3,000 to $5,000 is sufficient to purchase 72 optical crosspoints, plus 9 wavelength selectors and converters. This will only occur when advances in optical component technology allow integration of many optical devices within a single integrated component. While this level of integration is still years from realization, there do not appear to be any fundamental technical obstacles that would make it infeasible.

Figure 3 shows how the number of the different types of components grows with the number of external links, on a per channel basis (note the logarithmic scale on the x-axis). The chart shows that larger values of d are preferable, since wavelength selectors and converters are substantially more expensive than crosspoints. In general, if we hold n fixed, the number of stages shrinks, as d increases. Thus, for large d, the number of wavelength selectors and converters drops, while the number of crosspoints increases. The minimum system cost is obtained when d is roughly equal to the ratio of the cost of a selector-converter pair to the cost of an optical crosspoint.

4 Bust Switch Control

The detail at the right of Figure 1 shows how the control section of Burst Switch Elements (BSE) making up the multistage interconnection network can be implemented. In this design, the control section consists of a d port *ATM Switch Element* (ASE), a set of d *Burst Processors* (BP), and a *Burst Storage Manager* (BSM). The ASE switches arriving BHCs to the proper BP. Each BP is responsible for handling bursts addressed to a particular output link. When a BP is unable to switch an arriving burst to a channel within its output link, it requests use of one of the BSU's storage locations from the BSM, which switches the arriving burst to an available storage location (if there is one). Communication between the BSEs and the BSM occurs through a local control ring provided for this purpose.

The BHCs contain an *offset* field specifying the time from the arrival of the BHC to the time when the first bit of the burst is to arrive. It also contains a *length* field specifying the time duration of the burst. The BP uses this timing information to schedule switching operations. To allow time for queueing within the control portion of the system, BHCs typically precede bursts by a time period in the neighborhood of 10 μs. However, a short data burst may last for less time than this (with 10 Gb/s channels, a 1 KB burst is just 1 μs long). If short bursts are to be handled with reasonable efficiency, the control system cannot assign resources to bursts starting at the time the BHC arrives, but only during the time period that the burst will actually use it. That is, the control system must project future resource availability and schedule the use of resources in the future. This requires new *lookahead resource management* mechanisms.

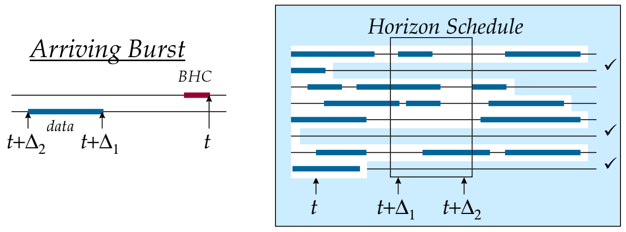


Figure 4: Horizon Channel Scheduling

4.1 Horizon Channel Scheduling

The primary function of the BPs is the assignment of arriving bursts to outgoing channels. For systems supporting channel rates of 2.4 Gb/s (or higher), the channel assignment must be done in less than 200 ns. One simple technique for fast channel assignment is *horizon scheduling.* In horizon scheduling, the BP maintains a time horizon for each of the channels of an outgoing link. The horizon is defined as the earliest time after which there is no planned use of the channel. The horizon scheduler assigns arriving bursts to the channel with the latest horizon that is earlier than the arrival time of the burst, if there is such a channel. If there is no such channel, the burst is assigned to the channel with the smallest horizon and is diverted to the BSU where it is delayed until the assigned channel is available. Horizon scheduling is straightforward to implement in hardware, but because it does not keep track of time periods before a channel's horizon when the channel is unused, it cannot insert bursts into these open spaces.

Figure 4 illustrates the operation of a horizon scheduler. The diagram shows an arriving burst that needs to be assigned to an outgoing channel. The burst header cell arrives at time t with the start of the burst arriving at $t+\Delta_1$ and the end of the burst arriving at $t+\Delta_2$. The right hand part of the figure shows bursts that arrived earlier that were assigned to channels by the horizon scheduler. The thick lines indicate time periods during which the various channels are scheduled to be in use and the unshaded region shows those time periods that are currently unavailable to new bursts. The box in the center highlights the time period during which the arriving burst will need an outgoing channel and the check marks at the right indicate those channels that it could be assigned to. Since the horizon scheduler prefers the viable channel with the latest horizon, it will select the bottom channel.

In some common situations, the horizon scheduler can provide good performance. Let $b_1, \ldots, b_n$ be a sequence of bursts, where b_i is characterized by a triple (r_i, t_i, ℓ_i); r_i is the time at which the BP receives the burst header cell informing it of the imminent arrival of the burst, t_i is the arrival time of the burst and ℓ_i is the length (time duration) of the burst. For convenience, assume that for $i < j$, $r_i \leq r_j$; that is, the bursts are listed in the order in which the burst header cells arrive. Define the *width* $W(B)$ of a burst sequence B, to be the size of the largest subset of bursts which all overlap in time with one another (that is, the earliest burst ending time in the set is later than the latest burst starting time in the set). A sequence of bursts B can be scheduled without delaying any burst if and only if the number of channels on the link is at least equal to $W(B)$. We would like to have a link scheduling algorithm that would schedule a sequence of bursts without delaying any burst, so long as $W(B)$ is no larger than the number of available channels.

A horizon scheduler can schedule a burst sequence $B = \{b_1 = (r_1, t_1, \ell_1), \ldots, b_n = (r_n, t_n, \ell_n)\}$ using no more than $W(B)$ channels if the bursts arrive in the same order as the burst header cells. However, we can allow some misordering of bursts and still achieve this level of performance. In particular, the horizon scheduler uses at most $W(B)$ channels if for all $i < j$, $t_i < t_j + \ell_j$. That is, we get good performance if no burst b_j precedes another burst b_i with $i < j$ by more than the length of b_j. In a burst switching system that does not provide burst storage, this condition can usually be satisfied, making horizon scheduling a good approach for such systems. On the other hand, the condition can be violated in systems that use storage to reduce the probability of discarding bursts.

4.2 Horizon Scheduling with Reordering

The above observations suggest a variant of horizon scheduling that performs better in more general situations. Rather than process bursts as soon as their burst header cells arrive, one can delay the scheduling of bursts, and then process them in the order of expected burst arrival, rather than the order in which the burst header cells arrive. Essentially, as the burst header cells arrive, we insert them into a resequencing buffer, in the order in which the bursts are to arrive. A horizon scheduler then processes requests from the resequencing buffer. The processing of a request is delayed until shortly before the burst is to arrive, reducing the probability that we later receive a burst header cell for a burst that will arrive before any of the bursts that have already been scheduled. More precisely, we define a parameter Δ and schedule a burst with arrival time t_i at time $t_i - \Delta$. Resequencing buffers developed for ATM switching can be used for this purpose. See, for example, references [3, 6].

By processing the requests out-of-order, we can get good performance, for a much wider class of burst sequences than we can with an ordinary horizon scheduler. In particular, we can schedule a sequence $B = \{b_1 = (r_1, t_1, \ell_1), \ldots, b_n = (r_n, t_n, \ell_n)\}$ using no more than $W(B)$ channels so long as for all $i < j$, either $t_i < t_j + \ell_j$ or $r_j \leq t_i - \Delta$; that is, the request for burst b_j arrives before burst b_i is scheduled. So the only situation in which extra channels are used is when $t_i \geq t_j + \ell_j$ and $r_j > t_i - \Delta$; that is, when the BHC for b_j arrives after b_i has been assigned to a channel and b_j is completed before b_i arrives. This cannot happen, so long as Δ is smaller than the shortest time period between the arrival of a BHC and the completion of the corresponding burst.

While out-of-order scheduling of bursts allows us to improve on the performance of horizon scheduling, it has an unfortunate side-effect. Since the scheduler does not assign bursts to channels until shortly before the bursts are to go out, we cannot know if a burst will be accepted or not until long after the burst header cell's arrival. In some situations, this information is needed in order to make other resource allocation decisions. What is needed, in these situations, is a way of deciding ahead of time, *when* a burst can be transmitted, while delaying the actual assignment of the burst to a particular channel.

4.3 Split Processing of Bursts

If we split the processing bursts into separate *burst scheduling* and *channel assignment* steps, we can get the advantages of out-of-order channel assignment while still allowing other resource management decisions to proceed. In split processing, the BP first decides when an arriving burst should go out. It then makes an entry for that burst in a reordering buffer which is processed in the order in which the bursts are to be sent to the output link using horizon scheduling. This approach allows us to schedule bursts using a minimum number of channels, but also enables other resource allocation decisions to be made at the time a burst header cell arrives, rather than delaying them until the burst is about to be forwarded. This is critical in multistage networks, where it is important to forward burst scheduling requests from stage to stage when the request first comes in, so that the downstream BSEs can allocate the resources they need to accommodate the burst.

To implement the burst scheduling step, we use a data structure that represents the *link usage curve*, which specifies the number of channels that are scheduled to be in use at every future time. When a burst header cell arrives, the algorithm consults the link usage curve to determine if the link is completely occupied at any time between the arrival of the burst and its completion. If not, the burst can be forwarded directly to the output, as soon as it arrives. Otherwise it will have to be delayed. The link usage curve is represented with a form of 2-3 tree [2] called a *differential search tree*, as illustrated in Figure 5.

Even with split processing, we still need a mechanism to decide when to schedule the transmission of a burst that must be delayed. Ideally, if an arriving burst must be delayed, we would like to insert it into the schedule at the earliest possible time. That is, we need to find the earliest time interval that is at least as long as the arriving burst duration and during which the buffer usage curve is always less than the number of channels on the link. It is possible to find this earliest time interval using the differential search tree, but it is difficult to do it quickly. Indeed, in the worst-case one may have to examine the entire search tree in order to find the earliest time at which the burst will fit. Moreover, there does not appear to be an alternative data structure that would allow an arriving burst to be quickly mapped to the earliest schedule gap that could accommodate it.

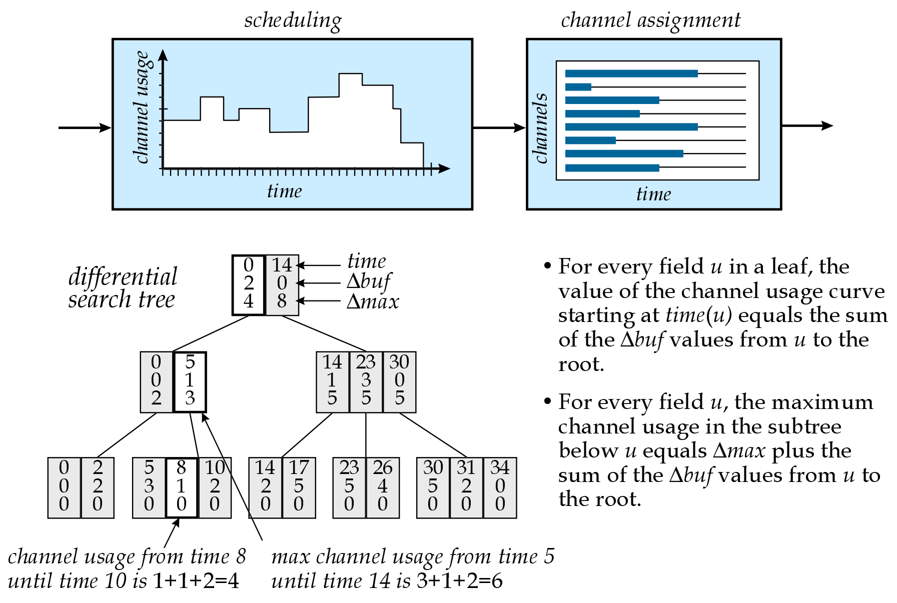
A simple alternative is to maintain a single horizon variable for the entire outgoing link. The value of this variable is the *latest* time when the link usage curve is equal to the number of channels on the link. When a burst arrives, we first use the differential search tree to determine if it can be scheduled for transmission without delay. If it cannot, we schedule it for transmission starting at the time given by the horizon variable. Since we know that the link is never completely used following the horizon value, it will always be possible to schedule the burst starting at that time. We then update the differential search tree and the horizon value.

This approach to scheduling bursts for transmission can produce suboptimal results if there are multiple time intervals where all the channels are in use, separated by time intervals when not all the channels are in use. Better performance can be obtained if the scheduling algorithm keeps a separate record of some small constant number of maximal time intervals during which the link is not fully occupied. This allows scheduling of arriving bursts in these *non-full periods*. So long as the number of such periods that must be tracked is kept small, a hardware scheduler can do the necessary matching operations in parallel, allowing a fast and practical implementation.

5 Closing Remarks

A small experimental burst switching system is currently under development at Washington University. This system will support seven external links with 32 channels of 1 Gb/s each. The primary purpose for this prototype is to demonstrate the control mechanisms needed to manage systems of this type. To provide maximum experimental flexibility, the Burst Processors in the prototype are being implemented with field programmable logic arrays (FPGA). In the first phase of the project, we are implementing a simple horizon scheduler, but as the project progresses more sophisticated algorithms will be incorporated and evaluated.

The development of burst switching systems that can be used in real networks must await substantial improvements in optical component technology. While there is still a long way to go, our analysis of the complexity of scalable burst switch architectures gives grounds for optimism that opti-


scheduling
channel usage
time
channel assignment
channels
time
differential search tree
time
Δbuf
Δmax
channel usage from time 8 until time 10 is 1+1+2=4
max channel usage from time 5 until time 14 is 3+1+2=6

- For every field u in a leaf, the value of the channel usage curve starting at $time(u)$ equals the sum of the Δbuf values from u to the root.
- For every field u, the maximum channel usage in the subtree below u equals Δmax plus the sum of the Δbuf values from u to the root.

Figure 5: Split Processing Algorithm for Channel Scheduling

cal switching technology may one day play a larger role in high performance routing switches. We believe that with the right combination of optical data path and electronic control, it should be possible to construct routing switches with very high capacities and the sort of sophisticated routing and resource management features needed for modern information networks.

References

[1] Chaney, T., J. A. Fingerhut, M. Flucke, J. Turner. “Design of a Gigabit ATM Switch,” *Proceedings of Infocom*, April 1997.

[2] Cormen, Thomas, Charles Leiserson, Ron Rivest. *Introduction to Algorithms,* MIT Press, 1990.

[3] Henrion, M. “Resequencing System for a Switching Node,” U.S. Patent #5,127,000, 8/90.

[4] Masetti, Francesco. “System Functionalities and Architectures in Photonic Packet Switching” In *Photonic Networks*, Giancarlo Prati (editor), Springer Verlag 1997.

[5] Siemens Semiconductor Group. “Parallel Optical Link (PAROLI) Family,” http://w2.siemens.de /semiconductor/products/ 37/3767.htm, 1998.

[6] Turner, Jonathan S. “Data Packet Resequencer for a High Speed Data Switch,” U.S. Patent #5,339,311, August 1994 and U.S. Patent #5,260,935, 11/93.

[7] Turner, Jonathan S. “Terabit Burst Switching,” Washington University Technical Report, WUCS-98-17, 1998.

[8] Turner, Jonathan S. “WDM Burst Switching,” *Proceedings of INET*, 6/99. Also, on the web at http:// www.arl.wustl.edu/~jst/pubs/inet99/inet99.html.