NETWORK PROCESSOR DESIGN FOR OPTICAL BURST SWITCHED NETWORKS

PronitaMehrotra[1], *IliaBaldine*[2], *DanStevenson*[3], *PaulFranzon*[4]

North Carolina State University (1,4) and MCNC (2,3)
Department of Electrical and Computer Engineering,
Box 7911, Raleigh, NC 27695-7911
{pmehrot,paulf}@ncsu.edu, {stevenson,ibaldin}@mcnc.org

ABSTRACT

Scalable hardware architectures are required for optical burst switched (OBS) where future fibers may be handling 4Tbps or more. Issues investigated include centralized vs. distributed architectures, scaling issues related to performance, and the hardware impact of Just-in-Time (JIT) vs. Just-Enough-Time (JET) signaling.

I INTRODUCTION

This paper discusses the design of a low-level network processor for use in Optical Burst Switching (OBS) over dense Wavelength Division Multiplexed (dWDM) networks. By low-level it is implied that the network processor handles only basic operations like forwarding, scheduling etc.

For an optical burst switched system, signaling is done out-of-band, and only the signaling channel goes through O/E/O conversion [1]. The data, or the burst, cuts through all the nodes from the source to the destination without going through any O/E/O conversion. The signaling channel carries messages related to the bursts that are soon to follow, and the network processor in our case parses and processes these messages. For this reason, we refer to the network processor as the message engine in the rest of the paper. The message engine supports the just-in-time (JIT) signaling protocol [2], where the bursts are transmitted by the source after a pre-determined delay, without waiting for any confirmation, in order to avoid the round-trip set-up delay. Several factors influenced the design of the message engine and these have been discussed where necessary.

The paper is organized as follows. Section II gives a brief overview of the JIT signaling protocol. Section III discusses some of the factors that influenced the design of the message engine and Section IV presents the architectural overview of the message engine. Finally, Section V lists the conclusions.

II JUST-IN-TIME SIGNALING PROTOCOL

Figure 1 shows the JIT signaling scheme for a single burst. The source (calling host) initiates the signaling by sending a SETUP message to its calling switch. The calling switch responds with a SETUP ACK message to indicate that a path to the destination is being established. Among other things, the SETUP ACK message carries a delay parameter, which tells the calling host to start transmitting the burst after that time. The calling switch also sends the SETUP message along the path and a channel is reserved at each of the intermediate nodes in the path. When the destination gets the SETUP message, it responds with a CONNECT message that goes back all the way to the source. The source doesn't have to wait for the CONNECT message before transmitting the burst, it can start transmitting the burst after the time specified in the SETUP ACK message from the calling switch. The source can maintain the connection by periodically sending the KEEP ALIVE message to keep the connections at the intermediate nodes in the valid state. The connection can be explicitly torn down by sending a RELEASE message.

III ISSUES IN THE DESIGN OF THE MESSAGE ENGINE

The primary functions that the message engine has to perform are:

- Determine if output fibers are available for each burst, and if so, which one is the most suitable;
- Generate messages; and

0-7803-6741-3/901/$10.00 © 2001 IEEE

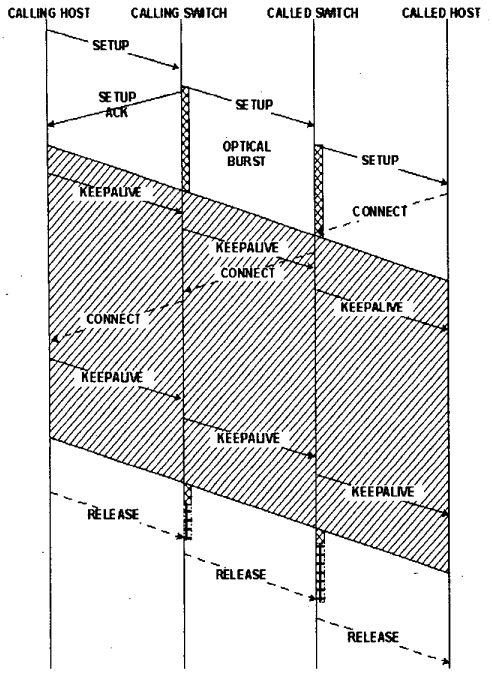

Figure 1: *Just-In-Time signaling scheme*

- Determine the settings for the all-optical crossbar at the heart of each switch.

The trend in recent network processors has been towards pushing intelligence and decision making into line cards as opposed to having a central unit process all messages. This distributed model with separate message engines on each fiber or port is necessary for a higher capacity. Furthermore, the message engine is pipelined in order to increase throughput. A centralized unit is simpler but severely limited in its message handling capacity especially when compared to the increasing link rates [3]. Figure 2 shows the burst handling capability per channel (where a channel corresponds to a particular wavelength on a particular port) of a centralized unit vs. that a distributed system for 50 and 100 wavelengths per fiber. The figure assumes that a pipeline cycle time of 70ns and 16 ports in the switch. Currently, the number of wavelengths for dWDM systems is up to 256 and in the future as the number of wavelengths per fiber keeps on increasing, there would be a higher demand on the processing power. The distributed system can handle 16 times more messages than the centralized system. The downside is that it needs as much more hardware to achieve this capacity.

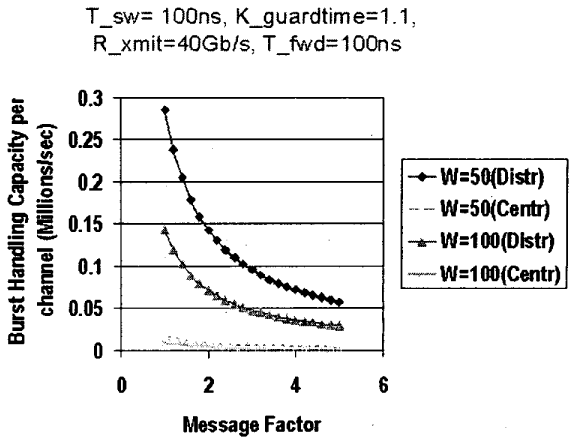

Figure 2: *Message Handling capacity for Centralized vs. Distributed Architectures*

The model used in the following figures assumes a worst-case scenario where all the input ports are fully loaded with minimum-sized bursts. The minimum size of the burts is determined by equating the message-processing rate to the burst transmission rate. Given the lack of any data indicating realistic burst sizes, the minimum burst size (MBS) provides a good metric and is useful for bounding design tradeoffs.

The model is constructed as follows:

$$MBS = \frac{(T_{fwd}.W.MF - T_{sw})R_{xmit}}{K_{guardtime}} \quad (1)$$

where, the message factor (MF) is the average number of messages that the message engine has to process for each burst (usually 2); the switch setup time (T_{sw}) is how long it takes to reconfigure the optical crossbar; T_{fwd} is the pipeline cycle time; R_{xmit} is the data transmit rate per wavelength; W is the number of wavelengths per fiber; and $K_{guardtime}$ is the design guard time around the burst transmission time as a factor of the total time. A lower minimum-burst size requirement is desirable because it places lesser constraint on the network and implies a lower rejection ratio.

The most important factor is the pipeline cycle time (T_{fwd}) in the processing of messages and keeping this as low as possible is crucial. Figure 3 and Figure 4 show how this affects the burst handling capacity per port and the minimum size requirement for bursts. The effective pipeline cycle time can be made small by using parallel DRAMs.

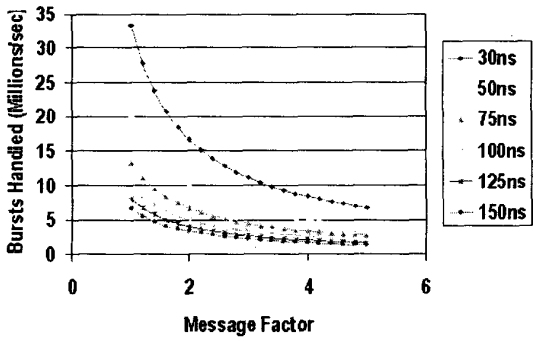

Figure 3: *Burst Handling Capacity for different pipeline cycle times*

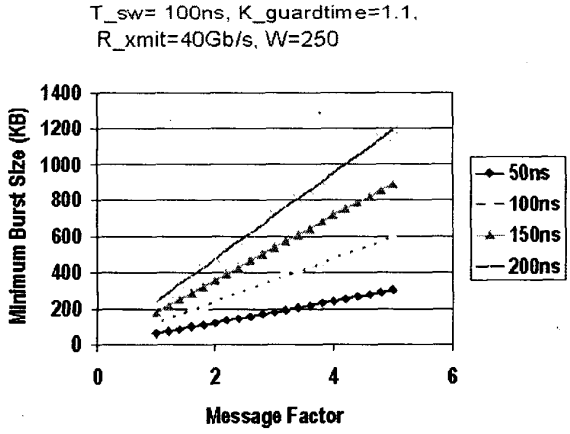

Figure 4: *Minimum Burst Size for different pipeline cycle times*

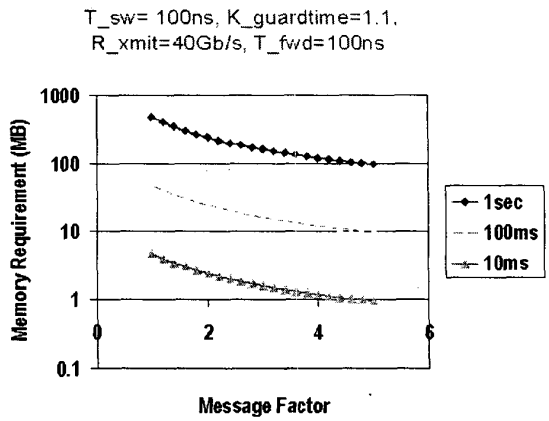

Figure 5: *Memory Requirement for scheduling in Just-Enough-Time Scheme*

The next issue in the design of routers for burst switched networks is the scheduler, or the output port request arbiter, which determines the available wavelength on the outgoing port and confirms whether the request can be accepted or not. Two scenarios exist for the function of the scheduler. In the first case, the arbiter configures the cross-connect immediately upon receiving the SETUP request. It releases the connection only after the burst has been transmitted or when an explicit release message has been received. In the second case (as in the Just-Enough-Time signaling [4]), the connection is set up for a certain amount of time in the future. The cross-connect in this case is not configured immediately upon receiving the SETUP message, but after a delay specified in the signaling message. Though this scheme makes better use of bandwidth, a large amount of memory is required to maintain the state of the switch at different intervals. The memory has to be organized with one entry per potential reservation so that it can be searched in a single cycle. One table would be needed for each wavelength on each output fiber. Figure 5 shows the amount of memory that would be required to maintain this information. A $75\mu s$ granularity is assumed in the assignment, and the longest time horizon for making a reservation is given on the right side of the chart. Given the latency that exists in optical networks, a realistic time horizon would be between 100ms and 1s. From the memory consumption point of view, maintaining a reservation table is clearly a more expensive option.

Our design, therefore, makes use of the first scenario where, a very simple timing information is maintained

to indicate whether a port is available or not when a SETUP message is received. This makes the design of the output port request arbiter simple and requires a fairly small amount of memory.

IV MESSAGE ENGINE IMPLEMENTATION

The specific functions that the message engine performs are as follows:

- Buffering the input messages
- Parsing the messages and determining the type of message. Additionally, performing a CRC on the message and generating a drop request when the CRC does not match
- Doing a forward lookup to determine the possible outgoing ports depending on the destination address. Multiple output ports are determined if possible, to ensure an overall higher network utilization
- Passing a request to the output ports to determine which of these is available
- Configuring the cross-connect if a request is accepted
- Generating an ACK/NACK depending on whether the set-up request is accepted or not

Figure 6 shows the micro-architecture of the message engine. The critical parts in the design are the forward lookup engine, the synchronizer and the output port requester/arbiter unit and these are discussed next.

The forwarding engine determines the output ports that the message needs to be forwarded to depending on the destination address in the message. For the OBS network under consideration, the addressing scheme is hierarchical and within a hierarchy the forwarding is similar to IP networks. This means that a longest-matching-prefix needs to be determined from the entries in the forwarding tables. For any node in the network multiple routing tables can exist for different hierarchies. Determining the output port requires memory lookups which can be very expensive especially if a DRAM is used. A random access to a DDR (Double Data Rate) DRAM takes around 60-70ns [5]. To obtain a faster T_{fwd}, multiple, parallel tables would be required.

The synchronizer is required in the design for the following reason. Consider the case when a set-up message is sent and a path is established from the source to the destination. For long bursts, the path is maintained by periodically sending a KEEP ALIVE message, which has the effect of maintaining the connection at each of the intermediate nodes for an additional time. If between the set-up and the keep-alive message or between two keep-alive messages, the forwarding table gets updated, then the subsequent keep-alive messages would be sent on a different path and the current path would be lost. The synchronizer maintains coherency between the keep-alive messages and the path they refer to. It does this by maintaining another table and referring to it instead of the forwarding table in determining the output port.

The output port requester/arbiter unit simply maintains information about which ports are busy and which are not. If a port is available and a set-up is requested on that port, it is marked busy and a corresponding message is sent to the switch controller to configure the cross-connect. This simple implementation of the scheduler does not require maintaining large amount of state information and therefore uses a fairly small amount of memory.

V CONCLUSIONS

Some of the architectural tradeoffs in the design of a message engine for use in optical burst switched environment were discussed. To limit the signaling channel to a single wavelength per fiber, a distributed high performance design is needed. In turn, the size of bursts that can be handled is limited by the pipeline cycle time of the message engine. If the cycle time is limited by DRAM access ($\approx 70ns$), then only consistently large bursts (several 100kB) can be handled. If OBS networks have to mainly carry smaller bursts, then parallel DRAMs will be needed.
A Just-in-Time signaling protocol is presented. The primary advantage of a Just-in-Time protocol over a Just-Enough-Time protocol, is a vast reduction in reservation memory needed.

VI ACKNOWLEDGEMENTS

We would like to thank the following funding sources: ARDA under contract MDA904-00-C-2133, and NSF under contract EIA-9703090.

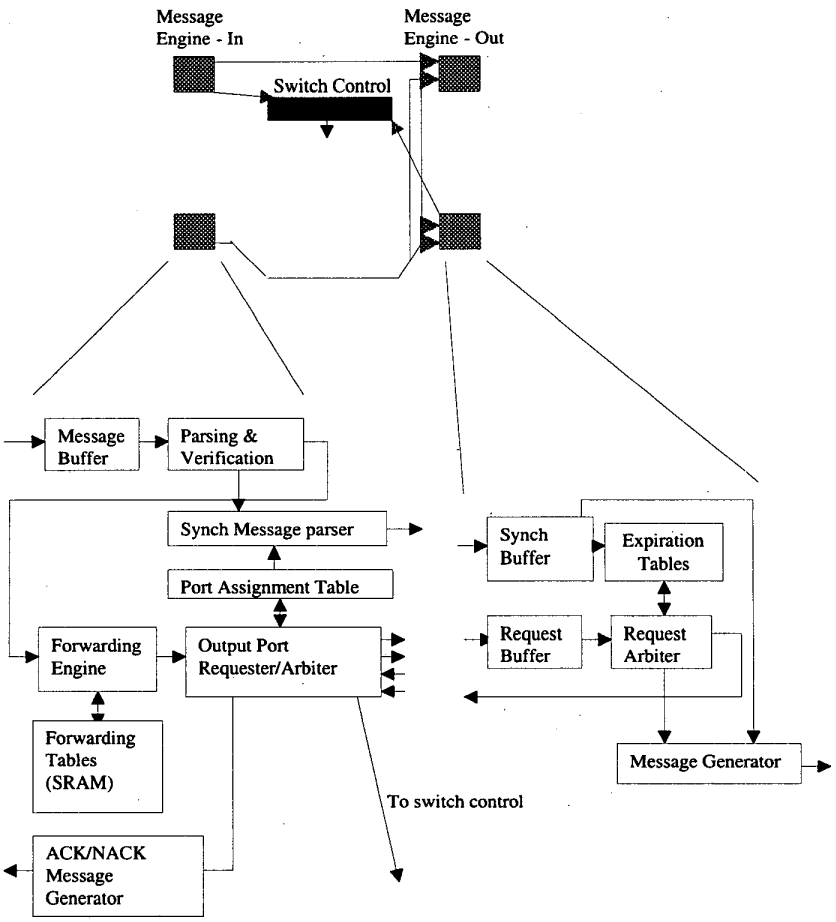

Figure 6: *Microarchitecture of the Message Engine*

REFERENCES

[1] M. Listanti, V. Eramo, and R. Sabella, "Architectural and Technological Issues for Future Optical Networks," *IEEE Communications Magazine*, vol. 38, pp. 82–92, Sept. 2000.

[2] J. Y. Wei and R. I. McFarland, "Just-in-Time Signaling for WDM Optical Burst Switching Networks.".

[3] N. McKeown, "Scalability of IP routers," in *Optical Fiber Communication Conference*, Mar. 2001.

[4] C. Qiao and M. Yoo, "Optical Burst Switching (OBS) - a new paradigm for an Optical Network," *Journal of High Speed Networks*, vol. 8, pp. 69–84, 1999.

[5] "128Mb DDR SDRAM Datasheet." (http://www.micron.com/products/datasheets/ddrsdramds.html).