Add Drop Benes Network for Low-Power Scalable Burst Packet Switching

Chankyun Lee and J.-K. Kevin Rhee
Dept. of Electrical Engineering,
KAIST,
Daejeon, South Korea
{ck.lee | rhee.jk }@kaist.ac.kr

***Abstract*— Optical fiber switch add-drop functions are implemented to conventional Benes network. This paper introduces novel network architecture, simple algorithm for add drop Benes network with passive medium optical switches for power efficient optical network. An analysis of the number of elements of networks will follow.**

Keywords-** **Rearrangeably nonblocking networks, Benes Network, Green optical network, Passive optical network, Optical communications.

I. INTRODUCTION

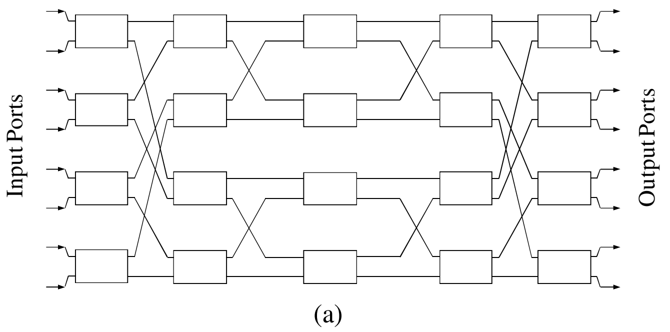
Benes network, which is a typical example of rearrangeably nonblocking networks, has been widely studied for a long time because of its unique characteristics – reconfiguration of some of switches are required to form new input/output pair connections and the total number of switch elements are far less than that of a strictly nonblocking network [1][2][3][4].

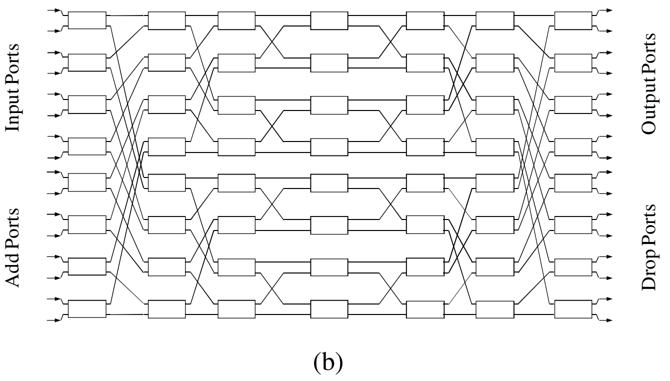
Conventional Benes network, as a bufferless switch, cannot handle the packet contention. Therefore, every different input port, in the same time slot, should be destined to a different output port. To resolve contentions, an add-drop function can be inserted to a Benes network. We will introduce ADBN (Add Drop capable Benes Network) with a little additional switch elements. The proposed architecture can be utilized with low-power fast passive-medium optical switches, such as an electro-optic switch, for peta-scale future internet applications. By combination of passive-medium switch Benes network with a burst packet switching, the network can overcome the power limit of peta-scale future internet.

II. ARCHITECTURE

Benes network consists of set of 2×2 switches. An $N\times N$ Benes network is composed by two $N/2\times N/2$ Benes networks in the inner stage and additional distribution networks in the outer stages, based on recursive configuration. Architecture of an 8×8 Benes network is shown in Fig. 1a. If the total numbers of input ports are N , required stages for Benes network are $2\log_2 N-1$ and each stage is consisted of $N/2$ switches. Therefore an $N\times N$ Benes network requires $N(\log_2 N-1/2)$ switch elements.

Add drop functions can be implemented in a conventional Benes network. A simple method to realize add-drop functions





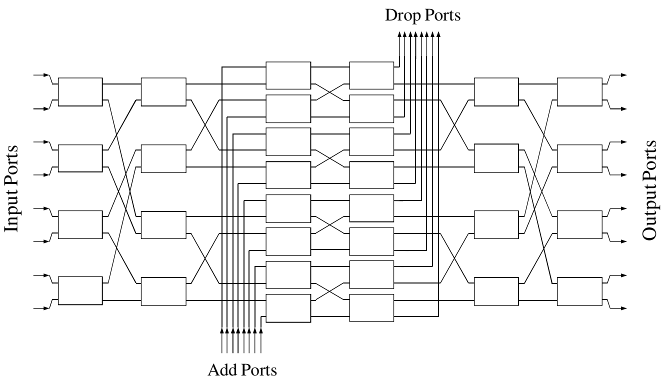


Figure 1. Architecture of convention Benes network (a), 16×16 Benes network for 8×8 add drop Benes network (b), and add drop Benes network (c) ($N=8$)

is using some input ports as add ports and some output ports as drop ports. Fig. 1b gives a schematic of an example to

This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the ITRC(Information Technology Research Center) support program supervised by the NIPA(National IT Industry Promotion Agency (NIPA-2009-C1090-0902-0006)).
This work was supported by the IT R&D program of MKE/IITA. [2008-F017-02, 100Gbps Ethernet and optical transmission technology development]

implement add-drop functions to Benes network by using $2N\times 2N$ Benes network. To make an 8×8 add drop capable Benes network, we can use a 16×16 Benes network, using eight input ports as add ports and eight output ports as drop ports. Using $2N\times 2N$ Benes network for an $N\times N$ add-drop Benes network requires $2N(\log_2 2N - 1/2)$ switch elements.

The suggested architecture of ADBN is shown in Fig. 1c, switches in middle stages of conventional Benes network are extended to four switch elements, to implement add-drop ports. The architecture of an ADBN network is based on the Benes network, under the recursive configuration. As the middle stage is extended to four 2×2 switches, the corresponded number of 2×2 switches in ADBN increases to $N(\log_2 N+1)$.

'Drop' function can be used for packet-drop to local networks and resolving the contentions in Benes network. Similarly, 'add' function is required for added packet from local networks or other switches. Due to the add-drop ports, add-drop functions can work with a normal switch operation, simultaneously.

III. Algorithm

We proposed the ADBN architecture which has a relatively small amount of additional 2×2 switch elements. Less number of switch elements per ports brings cost/power efficiency, however it also brings less freedom of the physical path from input/add ports to output/drop ports than that of conventional Benes network. Essentially, the Benes network requires an operation algorithm because it is not a strictly-nonblocking network. The low degree of freedom of the physical path requires a stricter algorithm than that of conventional Benes network for favorable operation of ADBN.

Fortunately, many algorithms for Benes network have been developed in the past [5][6][7]. Among them, one algorithm called looping algorithm is widely adopted because of simplicity and originality [5]. Likes Benes network itself, a looping algorithm also running under a recursive concept.

We suggest an algorithm for the ADBN, called ADLA (Add-Drop Looping Algorithm). The basic concept of ADLA is based on the looping algorithm. The ADLA consists of four steps - sorting, scheduling for non-contended packets, scheduling for drop packets, and scheduling for add packets. In the following, we will introduce ADLA step by step.

A. Sorting

Each packet has its own destination output ports. If more than two packets have same destination output ports, it will be considered as contention because one output port is available for only one input port. The ADLA sorts out contend packets and arranges them to the last. For example of ADLA for an 8×8 ADBN, packets are designed with an input port and a destination output ports as table I. In this example, the packets with input port 1 and 2, input port 3 and 4, and input port 6 and 8 are under the contention. An output port is available for only one packet, other packets are assumed as contend packet and have to be dropped. Result of sorting is shown in Table II.

TABLE I. Before sorting

Input	6	3	1	8	2	7	4	5
Ouput	3	7	6	3	6	5	7	1

TABLE II. After sorting

Input	6	3	1	7	5	8	2	4
Ouput	3	7	6	5	1	drop	drop	drop

B. Non-contended packet

After sorting the packets, the ADLA makes connection for the non-contended packets. Connections for non-contended packets are done by conventional looping algorithm. The looping algorithm determines switches states for connection from an input port to a destination output ports. Result of switches states after schedule non-contended packets is shown in Fig. 2a.

C. Drop packet

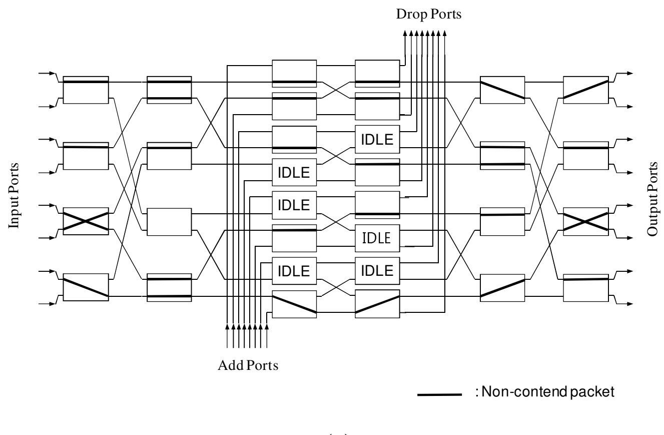
The packets can be dropped through $\log_2 N+1$ stage in the ADBN. When the total numbers of the packets to be dropped are n, n 2×2 switch elements in $\log_2 N+1$ stage will under the idle state. And thus, one of two 2×2 switches in $\log_2 N$ stage, which is connected with the idle state switch in $\log_2 N+1$ stage, also will under the idle state. ADLA schedules the contend packets to reach the idle state switch in $\log_2 N$ stage and determines the switch state in $\log_2 N$ stage to reach the non-idle state switch in the $\log_2 N+1$ stage. If all of two 2×2 switches in $\log_2 N+1$ stage which are connected with an idle state switch in $\log_2 N$ stage are under idle state, ADLA determines the state of switch in $\log_2 N$ stage to reach one of idle 2×2 switch in $\log_2 N+1$ stage randomly. State of non-idle switches in $\log_2 N+1$ is already determined by non-contend packets. Therefore, new packets are always connected to the drop ports. After schedules the drop packets, switches states are shown in Fig. 2b.

D. Add packet

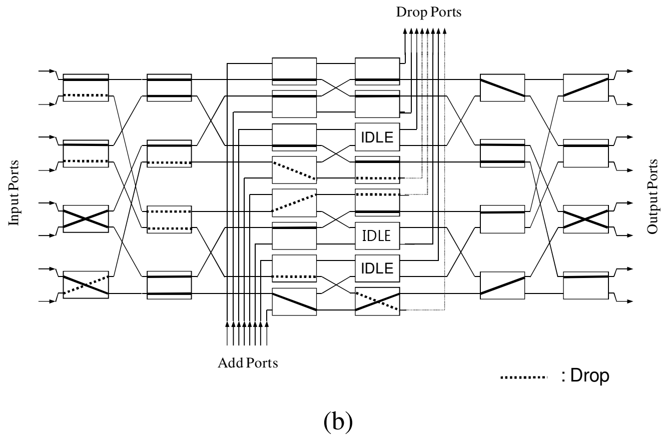
The packets can be added at the add ports. If a destination output port of the add packet is occupied by non-contended packet during the previews steps, add packet cannot be added. In this reason, it is needed to check whether the output ports of add packets are occupied or not, before schedules for the add packets. The odd orders of switches in the $\log_2 N+1$ stages can reach the $1\sim N/2$ output ports only. Similarly, $N/2+1\sim N$ output ports are available for even orders of switches in the $\log_2 N+1$ stages. Therefore, before schedules the add packets, it is needed to check destination output ports of add packet. If it is in the $1\sim N/2$, this add packet will be scheduled to reach the odd order idle switch in the $\log_2 N+1$ stages. If an add packet has the destination output port between $N/2+1\sim N$, it will be scheduled to pass the even orders idle switch in the $\log_2 N+1$ stage. Switches in the

$\log_2 N$ stage are selected automatically to reach $\log_2 N+1$ stage, add ports are also selected to reach the $\log_2 N$ stage.

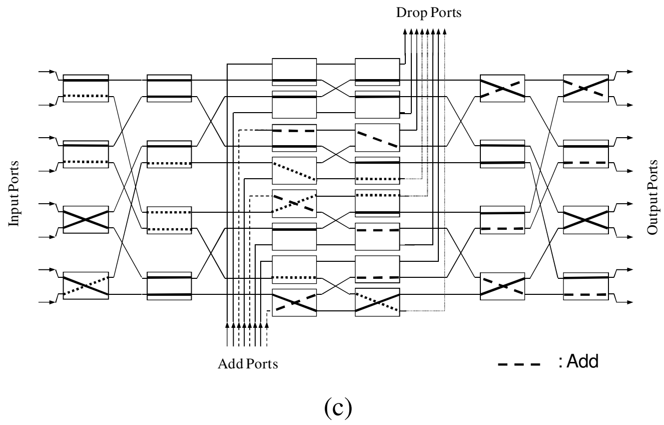
For example, assume that there are 4 packets wait to add, destination output ports are {8, 2, 3, 4}. But output port 3 is occupied by input 6. Therefore {8, 2, 4} can be added and only 8 is in the $N/2+1 \sim N$. By ADLA, it is scheduled to pass the even order idle switch in $\log_2 N+1$ stage and others for odd order idle switches. After the add function, the result of switches states are shown in Fig. 2c.



(a)



(b)



(c)

Figure 2. Switches states after step 2, for non-contended packets (a), step 3, for drop packets (b), and step 4, add packets (c) ($N=8$)

IV. Analysis

Section II explains some methods for add drop functionable Benes network – $2N\times 2N$ Benes network and ADBN. When compared with conventional $N\times N$ Benes network, the ADBN requires only 3/2 N additional 2×2 switches. This overhead is relatively very low when compare with that of $2N\times 2N$ Benes networks. The comparisons of number of 2×2 switch elements are shown in Fig. 3.

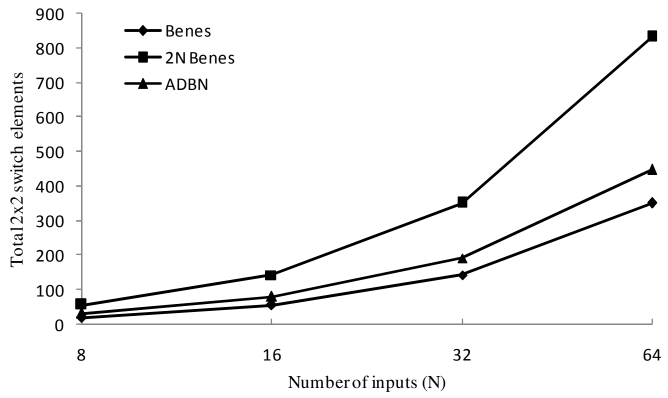


Figure 3. Comparisons of number of required 2×2 switch elements with various types of network – Conventional Benes network, $2N\times 2N$ Benes network to implement $N\times N$ add drop enable Benes network, and suggested ADBN.

V. Conclusion

In this paper, we suggest the noble architecture of add drop capable Benes network with small amount of additional 2×2 switch elements. For operation of the ADBN, simple algorithm called ADLA is introduced with examples. Power efficiency of ADBN is described by number of switch elements comparison of Benes networks and ADBN.

VI. References

[1] H. Scott Hinton, J. R. Erickson, T. J. Cloonan, F. A. P. Tooley, F. B. McCormick, and A. L. Lentine, “An Introduction to Photonic Switching Fabric,” Plenum Press, New York, 1993.

[2] C. Clos, "A study of non-blocking switching networks,” Bell Syst. Tech. J., vol. 32, pp. 406-424, Mar. 1953.

[3] V. E. Benes, “Permutation groups, complexes, and rearrangeable connecting networks,” Bell Syst. Tech. J., vol. 43, pp. 1619 – 1640, July 1964.

[4] V. E. Benes, “Optimal rearrangeable multistage connecting networks,” Bell Syst. Tech. J., vol. 43, pp 1641 – 1656, 1964.

[5] D.C. Opferman and N.T. Tsao-Wu, “On a class of rearrangeable switching networks-park I: Control algorithms,” Bell Syst. Tech. J., vol. 50, pp. 1579-1600, 1971.

[6] D. Nassimi and S. Sahni, “Parallel algorithms to set up the Benes permutation network,” IEEE Trans. Computers, vol. 31, no. 2, pp.148-154, Feb. 1982.

[7] K.Y. Lee, “On the rearrangeability of ($2\log N-1$) stage permutation networks,” IEEE Trans. Computers, vol. 34, no.5, pp. 412-425, May 1985.