2011 IEEE International Conference on High Performance Computing and Communications

C-switches: Increasing Switch Radix with Current Integration Scale

Juan A. Villar, Francisco J. Andújar, José L. Sánchez, Francisco J. Alfaro
Department of Computing Systems
University of Castilla-La Mancha
Albacete, Spain
Email: {juanan,fandujar,jsanchez,falfaro}@dsi.uclm.es

José Duato
DISCA
Technical University of Valencia
Valencia, Spain
Email: jduato@disca.upv.es

***Abstract*—In large switch-based interconnection networks, increasing the switch radix results in a decrease in the total number of network components, and consequently the overall cost of the network can be significantly reduced. Moreover, high-radix switches are an attractive option to improve the network performance in terms of latency, since hop count is also reduced. However, there are some problems related to the integration scale to design such single-chip switches. In this paper we discuss key issues and evaluate an interesting alternative for building high-radix switches going beyond the integration scale bounds. The idea basically consists in combining several current smaller single-chip switches to obtain switches having greater number of ports. This approach is independent of the evolution of single-chip switches and remains valid as integration scale keeps evolving. Simulation results show that with a correct internal switch design, this alternative achieves almost the same performance as single-chip switches with the same number of ports, which would be unfeasible with the current integration scale.**

I. Introduction

Interconnection networks are a key component for a wide range of multiprocessor systems, ranging from large supercomputers to multicore chips. High performance networks are essential in these systems, where high reliability in communications, high information transfer rates and very low latencies are critical. Often, the interconnection network is the subsystem that a more custom design requires. For instance, Tianhe-1A supercomputer [1], number one in the November 2011 Top500 list, is composed of standard processors and a fancy new interconnection network, which removes the interconnect bottleneck and significantly contributes to the high global performance of Tianhe-1A.

Interconnection network design is determined by the available technology. Recent advances on the technology have substantially improved the performance of the basic network components: links and switches. The latter are responsible for most of the interconnection network performance, and so they are the subject of major research. One of the main parameters characterizing network switches is the number of ports, which has a strong influence on cost, consumption and performance in the whole system.

Given a multiprocessor system with a large number of connected elements, increasing the number of switch ports results in a decrease in the number of switches and network links. As the cost of the network is proportional to the number of switches, it is clear that cost decreases by using switches with higher number of ports. Moreover, total consumption of the network is also considerably reduced as it is directly proportional to the number of switches in the network.

Regarding performance, in terms of latency, for example, it is clear that the use of switches with more ports involves a reduction in the average time to transfer data over the network. In particular, using fewer switches to connect the same number of elements reduces the number of hops and the number of possible packet collisions in the network, and thus the time to reach their destinations. Furthermore, having less switches, the total processing time of the packets in the switches along their paths is also reduced.

Thus, the design of switches with a high number of ports is an attractive option to improve the performance and reduce the cost of the interconnection network, especially for large multiprocessor systems. However, there are some problems to design such single-chip switches. One of them is related to the complexity of the switch logic. The switch becomes more complex as radix increases, taking up to a significant percentage of total system power [2]. The balance between cost and efficiency is not easy to work through, requiring a deep study regarding this trade-off. On the one hand, the size of some switch structures grows quadratically with the number of ports. That is the case of, for example, the aggregate buffer requirements as identified in [3], or the schedulers as stated in [4]. Moreover, traditional flow control policies are also affected by switch radix in two aspects [5]: the round trip time drastically increases, and the memories for storing flow control credits are linearly dependent on the round trip time. On the other hand, pin count will slowly increase next decade, according to the ITRS [6], and therefore switch ports number will slightly increase. Moreover, there are difficulties to apply some improvement techniques when the number of ports is high. For instance, Virtual Output Queuing (VOQ) [7] implementation becomes unfeasible in practice for switches

This work was supported by the Spanish MEC and MICINN as well as European Comission FEDER funds, under Grants "Consolider Ingenio-2010 CSD2006-00046" and "TIN2009-14475-C04", respectively; it was also partly supported by Junta de Comunidades de Castilla-La Mancha under grants "PEII11-0229-2343", and "POII10-0289-3724".

978-0-7695-4538-7/11 $26.00 © 2011 IEEE
DOI 10.1109/HPCC.2011.16

with large number of ports. To overcome these problems, different solutions have been proposed, but actually, they are postponing the problem for coming switch generations, because the switch size constraints are mainly determined by the current integration scale and package pin count.

To go beyond the integration scale bounds, an alternative solution for building high-radix switches is the combination of several low-radix switches. This solution has the advantage of obtaining higher number of ports. Moreover, some difficulties mentioned above lose importance.

The main idea is to implement m'-port switches from several smaller m-port switches. For instance, a m'-port switch consisting of two identical m-port switches ($m'/2 < m < m'$) can internally interconnect each other using $m - m'/2$ ports, using the remaining ports for external connections (Fig. 1). Note that this strategy makes possible to eventually go beyond integration technology and dramatically shorten the time-to-market. Moreover, this strategy will remain valid as integration scale keeps evolving.

An important consequence of this strategy for building larger switches is that the resulting switch will no longer be homogeneous. Switch performance will vary depending on the internal configuration. The internal switches interconnection can become a potential bottleneck if they have to support most of the traffic handled in the switch. Therefore, it is essential to minimize the impact of this bottleneck, otherwise the latency on the network will be increased.

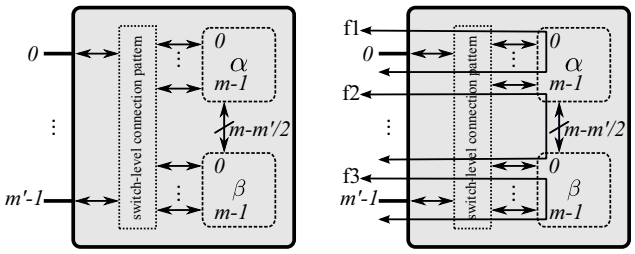
In this paper, we discuss key issues of this alternative for obtaining high-radix switches and evaluate its performance. Indeed, we show that the switch-level connection pattern[1] (SCP) and the communication bandwidth between internal switches are crucial for the switch behavior. A trade-off between both aspects is required to achieve efficient switch designs.

The paper is organized as follows: in Section II details about the proposed high-radix switch alternative are given. In Section III we obtain the optimal internal configuration of this kind of switches for a given network, and in Section IV we determine the minimum number of internal links to avoid the internal links become a bottleneck. In Section V we include some performance evaluation results, in order to check the availability of this alternative for building high-radix switches. Additional implementation aspects are provided in Section VI including a cost comparison. Section VII gives a brief review of existing proposals on high-radix switches. Finally, some conclusions are outlined in Section VIII.

II. High-Radix Switches by Combining Low-Radix Switches

As mentioned above, it is possible to build high-radix switches by combining several low-radix switches. This strategy makes possible to eventually overtake integration technology and dramatically shorten the time-to-market. Note that this will remain valid as integration technology continues evolving.

[1]We distinguish between *network-level connection pattern* and *switch-level connection pattern*. The former is the traditional interconnection pattern connecting switch-based networks (e.g., *butterfly* permutation in multistage interconnection networks); and the latter refers to how the ports of this kind of high-radix switch are mapped to the ports of the internal switches.



(a) Basic block diagram. (b) Optimal case.

Fig. 1. T-switch: diagram and possible situations.

This strategy opens a series of new issues that must be addressed so that it would be implemented in an efficient manner. In the next sections, we overview these issues that we have analyzed in [8] in a more formal way. Nevertheless, in this paper we provide an experimental quantification of their influence on network performance.

We define the Combined Switches and then we turn our attention to a particular subclass of this kind of switches, which will be used in order to show the characteristics and evaluate the performance of this high-radix switch alternative.

Definition 1. *A Combined switch, or simply C-switch, is a switch formed by several smaller interconnected single-chip switches (internal switches). The ports being offered by a C-switch are obtained from free ports of its internal switches after they are interconnected.*

This is a general definition because it does not specify either the number of smaller switches or their radix. Therefore any switch obtained by combining other smaller switches is included in this category. However, there exist some difficulties to build C-switches having many internal switches and a high heterogeneity degree.

In order to keep the internal latency low, a full-connected subnetwork interconnecting all the internal switches is preferred. As the number of internal switches increases, the number of ports of the internal switches dedicated to subnetwork connections grows as fast as the number of ports devoted to external communications decreases, so this way of building high-radix switches would lose interest. Therefore, it seems reasonable that the number of internal switches may not be too large.

On the other hand, a simpler internal C-switch design can be achieved if all the internal switches are equal. Although this aspect is not as restrictive as the number of internal switches, it would be also recommendable that all the internal switches have the same radix.

An interesting case is that where C-switches are built from only two identical internal switches. This subclass of C-switches still offers an important increase in the number of ports while the interconnection between the two internal switches is the simplest one.

Definition 2. *A Twin switch, or simply T-switch, is a switch formed by two identical smaller interconnected single-chip switches. The ports being offered by a T-switch are obtained from free ports of its two internal switches after they are interconnected.*

Considering that the two internal single-chip switches and the T-switch have radixes m and m', respectively, Fig. 1a shows a basic block diagram of a T-switch, where internal switches are denoted as α and β. Although T-switches seem to be simple, there are significant challenges in its design. Two of them stand out especially: (1) to obtain the appropriate SCP of internal subnetwork, and (2) to determine the adequate number of ports used to interconnect α and β switches.

Switch-level connection pattern has an important influence on packet latency. Time to cross the T-switch will be minimal when only one internal switch (α or β) is used (data flows f1 or f3 in Fig. 1b). The bad case is obtained when both internal switches are used for every path crossing the T-switch (data flow f2 in Fig. 1b). A more common situation is that where data flows f1, f2 and f3 in Fig. 1b coexist. In such situations, the main objective in the SCP design of T-switches (in general C-switches) is to minimize the use of the ports interconnecting internal switches (data flow f2). To obtain the best case is not trivial and an in-depth study is required, in which several factors, e.g., network topology, routing and traffic pattern must be considered. In [8] we show in a formal way how the optimal switch-level connection pattern can be obtained when these factors are considered.

Regarding the second challenge, the number of ports interconnecting the internal switches must be that which avoids the creation of an internal bottleneck between the α and β switches. It is clear that the greater the number of ports used to interconnect the internal switches the lower the probability of this interconnection becomes a bottleneck. However, as mentioned above, as the number of ports devoted to interconnect the α and β switches increases the T-switch radix decreases. Note that we are assuming that all the ports provide the same bandwidth (otherwise instead of using the number of ports, the aggregate bandwidth should be used). Consequently, a trade-off between the number of ports to interconnect internal switches and the SCP must be found.

In the following sections we analyze in detail these two key design aspects, taking into account that the aim is to find the optimal internal configuration of C-switches. Since the optimal configuration of the C-switches depends on the conditions under which they are used, e.g., network type, topology, routing algorithm, and traffic pattern, we use a specific network in our study.

We have chosen the k-ary n-tree BMIN network [9], a subclass of fat-trees, which is one of the most common topologies today in the largest supercomputers on the Top500 list. Regarding C-switches, we have used T-switches. We are considering T-switches for purely illustrative purpose that makes the formal development simpler and easier to understand, but the approach is general enough to be apply to any C-switch.

The routing algorithm is DESTRO [10], which is a deterministic routing algorithm for fat-trees. We have selected DESTRO because of its interesting features: simple hardware implementation, reduced routing time and in-order packet delivery. Moreover, it is able to evenly balance network traffic and reduce network contention. We have considered uniform traffic pattern because it is commonly used in network performance evaluations.

The following notation is used below:

- N is the total number of terminals, or end nodes.
- k is the T-switch arity, i.e., the number of ports that connect to terminals/switches in the previous stage, and to switches in the next stage (if available). The total number of ports of a $k \times k$ switch is $2k$. The ports faced to the previous stage are labeled from 0 to $k-1$, and the ports connected with the switches in the next stage are labeled from k to $2k-1$. We assume k is even and $k \geq 4$.
- n is the total number of stages, where $n = \log_k N$.
- $\langle s, o \rangle$ is a tuple that identifies uniquely a switch, where s refers to the stage ($0 \leq s < n$), and $o = o_{n-2} \ldots o_1 o_0$ indicates the position of the switch inside the stage, where $0 \leq o_i < k$ and $0 \leq i < n-1$.
- c is an unidirectional link.
- w is the link bandwidth.
- γ_c is the load on a link c that is defined as the ratio of the bandwidth demanded from link c to the bandwidth of the terminals.
- r is the total number of links interconnecting the internal switches of a T-switch.
- $R_{h,h'}$ is the set of paths between the terminals h and h'. card($R_{h,h'}$) represents the total number of paths in the set $R_{h,h'}$. A path $P \in R_{h,h'}$ is an ordered sequence of links needed to arrive from h to h'.

III. Switch-level Connection Pattern

The switch-level connection pattern (SCP) is a key design issue of combined switches, and it has an important effect on the switch behavior. In order to show that, in the following we obtain the optimal SCP for all the C-switches in a given network, applying the methodology described in Section III-A, and in Section V we study the influence of the SCP by evaluating the performance of the network considering different possible situations.

A. C-Switch Configuration Methodology

Our methodology for determining the optimal SCP of C-switches consists of the following main steps:

- Network paths analysis. The purpose of this step is to determine the connections required in each C-switch at network level and the number of times all these connections are used taking into account all the possible paths used by the packets. It can occur that some of the possible connections in the C-switches support one or more paths, and however, there may be connections that are never established.

- Switch classification. Depending on the network characteristics and load conditions, few or many different C-switch configurations could be obtained. In this step, C-switches are grouped according to their connection requirements, and so, several types of C-switch will be distinguished.
  For instance, in a fat-tree topology C-switches in different stages may require different SCPs, and the same may even occur with C-switches in the same stage. Therefore, several types of C-switch will appear. When a simple traffic pattern and balanced routing algorithm are used, it is likely all the C-switches in the network require the same SCP, thus only one type of C-switch will be considered.
- Switch configuration. From connection requirements and given the number of internal switches forming the C-switch, this last step consists in finding the optimal configuration for each class of C-switch. That is, we must find the optimal SCP of each class, trying to minimize the use of the interconnection between internal switches.

Summing up, given the network topology, the routing algorithm and the traffic pattern, we can determine the paths generated in the network, the C-switches used by each path, and the connections required in each C-switch. If the two ports involved in a C-switch connection are connected to the same internal switch, the paths using that connection will only use one internal switch when passing through the C-switch. Therefore, the objective of the methodology is to get an SCP where most of the connections satisfy this condition. In general, C-switches with different connection requirements will have different internal configurations, thus the methodology proposes to search for the optimal SCP for each C-switch, or group of C-switches requiring the same connections, separately. Finally, if possible, the same SCP for all the C-switches will be found.

The first two steps of this methodology are independent of the internal C-switch structure. Obviously, the third step depends on the structure of the C-switch.

Note that although the methodology is general and can be applied considering different C-switches and interconnection networks, as mentioned above the C-switch configuration depends on the particular network properties. Therefore, in order to apply this methodology, network type, topology, routing algorithm and traffic pattern must be specified.

B. Optimal SCP for all the Network Switches

In this section we obtain the optimal SCP for all the T-switches in the network described in Section II by applying the methodology in Section III-A. Since the SCPs have been derived from a formal analysis, we can state that the reached SCPs are optimal under the considered network characteristics. In order to keep the length of this paper short, we cannot include here the whole formal development, which is available in [8].

As described, taking into account the network topology, the routing algorithm, and the traffic pattern, we are able to know all the different paths among all the terminals in the network. We can also know the particular T-switches[2] being used by each path, and for each T-switch, the input and output port used for entering and leaving that T-switch. Therefore, given a T-switch, we can determine the required connections among all the ports, and calculate the number of times each connection must be established. Moreover, we can obtain this information distinguishing three different cases, which correspond with forward, backward, and turnaround connections in a BMIN.

TABLE I
NUMBER OF PATHS PASSING THROUGH A SWITCH IN A NETWORK UNDER UNIFORM TRAFFIC.

$$C_f(\langle s,o\rangle) = \begin{cases} k^{n+1}-k^{s+2}, & \text{if } s\in[0,n-2] \\ 0, & \text{if } s=n-1 \end{cases}$$

$$T(\langle s,o\rangle) = (k-1)k^{s+1}$$

$$C_b(\langle s,o\rangle) = \begin{cases} k^{n+1}-k^{s+2}, & \text{if } s\in[0,n-2] \\ 0, & \text{if } s=n-1 \end{cases}$$

$$C_{f\ 0\le l<k\ k\le l'<2k}(\langle s,o\rangle,l,l') = \begin{cases} k^{n-1}-k^{s}, & \text{if } s\in[0,n-2] \\ 0, & \text{otherwise} \end{cases}$$

$$T_{0\le l,l'<k\ l\ne l'}(\langle s,o\rangle,l,l') = k^s$$

$$C_{b\ k\le l<2k\ 0\le l'<k}(\langle s,o\rangle,l,l') = \begin{cases} k^{n}-k^{s+1}, & \text{if } s\in[0,n-2] \text{ and } l'=l-k \\ 0, & \text{otherwise} \end{cases}$$

We denote by C_f, T, and C_b the number of different paths passing through a T-switch in the forward, turnaround and backward directions, respectively, at switch level and port-pair level. For instance, $C_f(\langle s,o\rangle)$ is the number of paths passing through the T-switch $\langle s,o\rangle$, $0\le s<n-1$, in the forward direction; and $C_f(\langle s,o\rangle,l,l')$ is the number of times the connection between the ports l and l' is used, $0\le l<k$ and $k\le l'<2k$. It is obvious that both expressions are interrelated, so that the former is obtained from the latter when all the possible pairs of ports (l,l') are considered. Similar definitions are given for C_b and T expressions.

TABLE I includes the final values for all these expressions. In the following, we comment some details that are easily deduced from them:

- There are $k^{n+1}-k^{s+2}$ different paths passing through a T-switch in the forward and backward directions. It is true for any T-switch belonging to all the stages in the BMIN network except the last one.
- There are k^n-k^{s+1} paths in the backward direction passing through a T-switch using an input port l and an output port l', but only if $l'=l-k$ where $k\le l<2k$, and $0\le l'<k$.

[2]As mentioned when describing the methodology, the first and second steps are independent of the internal structure of T-switches, therefore we could use the term "switch" instead of "T-switch". However, for clarity, we will use T-switch in all the steps of the methodology.

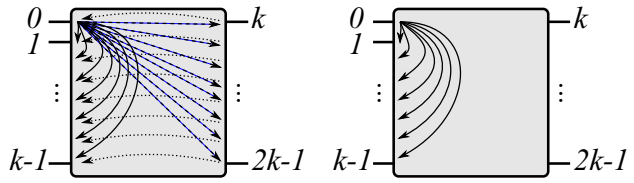


Fig. 2. Connections required for uniform traffic pattern on each T-switch of (a) Type A, and (b) Type B.

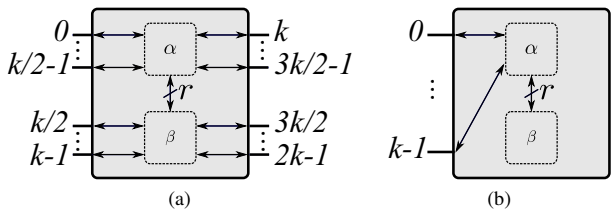


Fig. 3. Optimal switch-level connection pattern for each T-switches of (a) Type A, and (b) Type B.

- All the T-switches of the BMIN network establish $(k-1)k^{s+1}$ turnaround connections with independence of their position in the stage.

After completing the first step of the methodology, we know how the paths are passing through the T-switches in a fat-tree, so that we can search for the optimal SCP of all the T-switches in the network by using the obtained expressions in TABLE I.

The second step of the methodology divides all the T-switches in as many groups or types of T-switch as different connection sets have been found. Although the same switch configuration could be optimal for all the T-switches in the network, it is reasonable and easier to search for the optimal configuration for each group separately, and then trying to find that unique configuration.

Fig. 2 shows the two types of T-switches previously identified: all the T-switches are Type A (Fig. 2a) with the exception of those placed at the last-stage in the BMIN network. Thus, the T-switches of Type B (Fig. 2b) correspond with the last-stage T-switches. Notice that Fig. 2 shows only the paths passing through the T-switch using the input port zero in the ascending and turnaround phases. The same paths are obtained for the remaining input ports, but they are not shown in the figure for the sake of clarity. Note that the paths in the descending phase are independent of the other phases. Specifically, for uniform traffic pattern the port l, $k \leq l < 2k$, always connects with the port $l' = l - k$ in the descending phase.

Finally, the third step of our methodology consists in finding the optimal internal connection structure for each type of T-switch. For a given T-switch and a given connection between the ports l and l', we must try that l and l' are linked to two ports of the same internal switch. If this is possible for all the connections, we obtain the ideal or best SCP for that T-switch. When this is not possible, the optimal SCP for a given T-switch is that minimizing the number of times both internal switches are used in order to pass through the T-switch.

We can obtain the optimal SCP for each type of T-switch, and then try to find only one SCP being optimal for all the T-switches in the network. As we show in [8], in this case it is not possible to have an unique optimal SCP, and two different SCPs are needed (Fig. 3). One of them (Fig. 3a) is an optimal SCP for the T-switches of Type A, and the other (Fig. 3b) is an optimal SCP for T-switches of Type B in the network.

In Fig. 3a we see that T-switch ports are associated with the two internal switches such that ports from 0 to $k/2-1$, and ports from k to $3k/2-1$, are connected to the α switch, and the remaining ports to the β switch. It is important to note that other configurations are optimal, but all of them must satisfy that the ports l and $k+l$ are connected to the same internal switch. The SCP shown in Fig. 3a minimizes the number of paths using the r internal links when passing through T-switches of Type A. In [8] we prove that number is $\frac{1}{2}k^{n+1}$. The proof has not been included here in order to keep this paper short.

On the other hand, in Fig. 3b it can be seen that the k ports used in the last-stage T-switches (ports from 0 to $k-1$) are connected to the α switch. It is obvious that this configuration is optimal because the r internal ports used for interconnecting the α and β switches are never used. Logically, the same result is obtained if the k ports are connected to the β switch.

Since the traffic pattern is simple we obtain a few different types of T-switches, and optimal SCPs. However, when more complex and irregular traffic patterns were considered, more types of T-switches and SCPs would be expected.

IV. Number of Ports for Connecting Internal Switches

As mentioned above, the links interconnecting internal switches can become a bottleneck if insufficient bandwidth is considered. On the other hand, an excessive number of internal links produces an underutilization of resources without increasing network performance, and decreasing T-switch radix. It is essential to use an appropriate number of ports of internal switches for this interconnection. Next we determine that adequate number of ports, which allows to obtain the maximum network throughput.

Maximum network throughput is reached when traffic saturates some link [11], [12]. If no links are saturated it means that the network will not being working at maximum throughput, and then the network will admit more traffic. Thus, the link load is used to determine the maximum network throughput.

Under a particular traffic pattern, the link carrying the largest fraction of the traffic determines the maximum link load $\gamma_{\max}$. The network will reach the maximum throughput if the link load is equal to its bandwidth w. Any additional traffic will saturate the link. Therefore, the ideal network throughput is defined as the network bandwidth that saturates the link that becomes a bottleneck $\Theta_{ideal} = w/\gamma_{\max}$.

In particular, for uniform traffic and balanced routing function, the computation of $\gamma_{\max}$ can be simplified by determining the upper bound on maximum link load $\gamma_{\max,UB}$ that is the largest $\gamma_{c,UB}$ over all links: $\gamma_{\max,UB} = \max_{c \in C} \gamma_{c,UB}$,

where

$$\gamma_{c,UB} = \frac{1}{N} \sum_{h \in N} \sum_{h' \in N} \sum_{P \in R_{h,h'}} \begin{cases} 1/\text{card}(R_{h,h'}), & \text{if } c \in P \\ 0, & \text{otherwise} \end{cases}$$

In a BMIN network built with T-switches, the saturated links are either the links that connect the first-stage switches to the terminals, or the links interconnecting the internal switches of T-switches. Next, we reason which links become a bottleneck before:

- In the former, we have $1/\text{card}(h, h') = 1$ because the DESTRO routing algorithm guarantees $\text{card}(R_{h,h'}) = 1$ for every pair (h,h') of nodes in the network. To compute $\gamma_{\max,UB}$ we need to know what is the link supporting the highest number of paths. Then, we divide such a number by N so that the link load is per terminal.
  Under uniform traffic pattern, the links that carry most traffic, which are those connecting the first-stage switches to terminals, support a total of N paths, and therefore, $\gamma_{\max,1^{st}\text{stage links},UB} = N/N = 1$ and the throughput on link is $\Theta_{\max,1^{st}\text{stage links},UB} = w$.
- In the latter, by applying a similar reasoning, we determine the load on internal links by knowing that they are used by a total of $\frac{1}{2}k^{n+1}$ bidirectional paths (T-switches of Type A in Section III-B). The total number of paths used in each direction is $\frac{1}{4}k^{n+1}$.
  Assuming that the paths passing through a T-switch are equally distributed between the r links interconnecting its internal switches, we have

$$\gamma_{\max,\text{internal links},UB} = \frac{\frac{1}{4}k^{n+1}}{N\ r} = \frac{\frac{1}{4}k^{n+1}}{k^n\ r} = \frac{k}{4r}$$

Concluding, the saturated links are those that have the largest link load

$$\gamma_{UB} = \max \begin{cases} \gamma_{\max,\text{internal links},UB} \\ \gamma_{\max,1^{st}\text{stage links},UB} \end{cases} \Rightarrow \max\left\{\frac{k}{4r}, 1\right\}$$

$\frac{k}{4r}$ is greater than 1 if $r < \frac{k}{4}$. Consequently, we have

$$\gamma_{UB} = \begin{cases} \frac{k}{4r}, & \text{if } r < \frac{k}{4} \\ 1, & \text{otherwise} \end{cases}$$

and the throughput is

$$\Theta_{UB} = \begin{cases} \frac{4rw}{k}, & \text{if } r < \frac{k}{4} \\ w, & \text{otherwise} \end{cases}$$

Based on this result we know that links interconnecting the internal switches are no longer a bottleneck if r is equal to or greater than $k/4$. In that case, a BMIN network built with T-switches will behave like a BMIN network in terms of throughput.

V. Performance Evaluation

In this section we evaluate the potential of C-switches, again using the T-switches subclass. The evaluation has been driven by simulation and focused on the two key issues discussed before: switch-level connection pattern and bandwidth of interconnection between internal switches.

In the following subsections, we describe the simulation model, the different network configurations, and the switch-level connection patterns used in the experiments. Finally, we provide the simulation results and analyze them in detail.

A. Simulation Model

To perform the evaluation presented in this paper, we have used a detailed event-driven simulator able to model different kinds of switch-based networks and, of course, C-switches.

Switch-based interconnection networks cover a wide range of network configurations. To perform this evaluation, we have chosen a representative case which has been described in Section II. The k-ary n-tree topology has been selected.

DESTRO deterministic routing algorithm [10] has been implemented at network-level. In this evaluation study, due to the current simulation tool state, we have only considered 32-port T-switches. These switches are formed by two internal 24-port switches, such that 16 ports are used for external ports and 8 ports for interconnecting the two internal switches ($m = 24$ and $m' = 32$ in Fig. 1b). We have used 24-port single-chip switches because they are commercially available today (e.g., InfiniScale III [13]). Nevertheless, the simulation tool is currently being improved to support larger switches.

We have evaluated networks with 4096 end-nodes. Other assumptions are an input queued switch design with virtual output queuing at switch level; 1 GByte/s full-duplex link bandwidth; round-robin scheduler per output port; 2 KBytes packet size; every queue buffer has capacity of 64 packets, among others.

B. Case Studies

In order to evaluate the viability and potential of combined switches, we have estimated the performance of the network described in Section V-A, considering T-switches with the switch-level connection pattern that has been obtained in Section III to be the optimal for T-switches under the simulated conditions (this network will be referred to as O-BMIN in the figures). Then, we have compared the performance of this network with that offered by an equivalent network considering single-chip switches with the same radix as T-switches. This network will be referred to as U-BMIN and it represents the upper performance threshold.

Moreover, in order to show the importance of the switch-level connection pattern we also consider another case which represents an arbitrary and bad internal switch design, labeled as B-BMIN in the figures. Therefore, the O-BMIN and B-BMIN networks use T-switches with the optimal SCP and an arbitrary SCP, respectively. The optimal and arbitrary SCPs can be seen in Fig. 3 and Fig. 4 for the k×k T-switches, respectively. Also, we include a fourth case (L-BMIN): a network

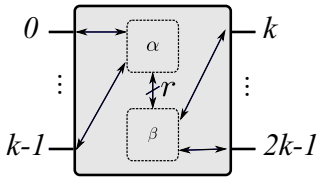


Fig. 4. Arbitrary switch-level connection pattern for all the types of T-switch under uniform traffic pattern.

connecting the same number of end-nodes, but using single-chip switches, which is feasible with the current integration scale.

On the other hand, the aim of this evaluation is also to know if the performance gap between the analyzed networks is significant. If so, we have also wanted to measure this gap.

Note that U-BMIN, O-BMIN and B-BMIN, are 16-ary 3-tree networks with a total of 768 switches of 32 ports. However, L-BMIN is a 8-ary 4-tree network using 2048 single-chip switches of 16 ports.

C. Performance Results

We have run two different tests. The first test has the aim of measuring the performance gap between the aforementioned case studies. The aim of our work is to evaluate and identify potential problems that limit the efficiency of T-switches (in general C-switches).

The figures of this section show the average values and the confidence intervals at 95% confidence level of thirty different simulations performed at a given input load.

Fig. 5 shows throughput and latency results for all the network configurations. Some interesting details can be observed:

- When optimal switch-level connection pattern is considered, the T-switch based network (O-BMIN) is able to offer similar performance to high-radix single-switch based network (U-BMIN). Therefore, combined switches are confirmed as a good alternative to build high-radix switches.
- However, the B-BMIN network reaches the saturation point at only 40% of normalized network load. The different performance obtained is due to the fact that with an arbitrary and bad SCP for T-switches much of the traffic needs an extra hop. So, it is also confirmed that it is essential to determine in each case the optimal switch-level connection pattern for combined switches.
- When U-BMIN and L-BMIN are compared, we have noticed that latency of L-BMIN is higher than latency of U-BMIN (e.g., 14% at the 50% of network load). This increase is due to L-BMIN has one more stage than U-BMIN, so the paths are longer.
- Also, we can see that the number of ports devoted for interconnecting the internal α and β switches does not produce any performance degradation. This verifies the conclusion that comes from the formal analysis.

In the first test, we have used eight ports of the internal switches for interconnecting each other. However, we need to set the number of ports for interconnecting the internal

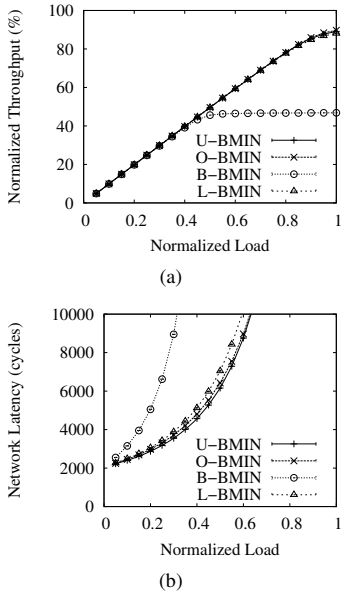


Fig. 5. (a) Throughput and (b) latency for the different configurations.

switches that are necessary to have enough bandwidth between the internal switches, without overdimensioning such a quantity. Otherwise, we may produce a degradation in external communication performance. To tune that, we have performed a second test that measures how the number of ports for interconnecting internal switches influences on the global performance.

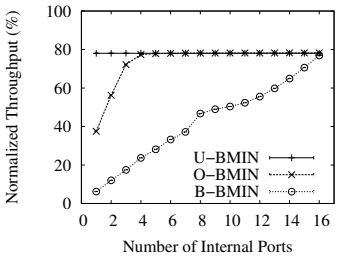


Fig. 6. Network throughput in function of the number of ports interconnecting the internal switches.

In Fig.6 we can see the network throughput results when network load is at 80% for all network configurations (except L-BMIN). In order to perform the second test, we have instrumented the simulator to vary the number of ports for interconnecting the internal switches from 1 to 16, while the number of external ports remained the same.

Similarly, we can notice more relevant details:

- We can observe that the switch-level connection pattern is the key factor that influences the global performance, and the number of ports is not so relevant.
- When the number of ports is four (it coincides with $k/4$ that comes from the formal analysis), the O-BMIN network achieves the same performance as the U-BMIN network. However, when the B-BMIN network is considered, the internal bottleneck does not disappear until the

number of ports interconnecting the internal switches is equal to the T-switch arity ($k = 16$).

- It is important to remind that in the results shown in Fig. 5 the number of ports interconnecting the internal switches is eight, but when the T-switch has the optimal switch-level connection pattern only four ports are enough to achieve the maximum performance. In such a case, these four extra ports could be used for external communication instead of devoting them for internal communication, and in this way, the switch radix would be effectively increased.

VI. Implementation Aspects

In this section, we determine the increment factor in the number of ports obtained by means of T-switches with respect to single-chip switches used to build them. Moreover, we estimate the implementation cost of BMINs build from this kind of switches and compare it with the cost of BMINs formed from single-chip switches similar to the internal switches used into T-switches. It is a simple cost estimation and a more detailed study would be required to know real values, but it is helpful for us in order to get an idea of the economic viability of these switches.

It must be noted that we consider here the same network characteristics as in previous sections. These are the k-ary n-tree BMIN network topology, the uniform traffic pattern, and DESTRO routing algorithm. Moreover, when $k \times k$ T-switches are considered, they will setup the optimal SCP and number of ports for interconnecting the internal switches, i.e., both α and β switches have a total of $k + k/4$ ports. On the other hand, we assume that the external links and links interconnecting the internal switches of T-switches are all bidirectional.

A. Radix Increment Factor

T-switches increase by 60% the number of ports of the internal single-chip switches. The calculation of this increment (Δ) is obvious:

$$\Delta = \frac{2k - (k + k/4)}{k + k/4} 100 = \frac{3}{5} 100 = 60\%.$$

We can get further increments if more internal single-chip switches are considered. However, as it was highlighted in Section II, if more internal switches are used new problems arise that must be solved, e.g., internal subnetwork complexity increases and average latency in the T-switch also increases.

B. Cost Estimation

In a nutshell, the network cost is divided between the switching elements cost, and the cabling cost. There are more expenses apart from those, but for the sake of clarity we do not take them into consideration. Nevertheless, the total network cost is proportional to the number of switches, cables, and their length. In this sense, we consider that the most significant expenses can be estimated by using the following cost metrics:

- Total number of single-chip switches (C).
- Total number of external links (E).

TABLE II
Values of the cost metrics.

Metric	U-BMIN	O-BMIN	L-BMIN
C	$C_U = \frac{N}{k} n$	$C_O = 2\frac{N}{k} n$	$C_L = \frac{N}{k'} n'$
E	$E_U = N \times n$	$E_O = N \times n$	$E_L = N \times n'$
I	$I_U = 0$	$I_O = \frac{N}{4} n$	$I_L = 0$

- Total number of internal links interconnecting the internal switches (I).

In the following, we calculate the value of these metrics for each case study: U-BMIN, O-BMIN, and L-BMIN. The B-BMIN case differs from the O-BMIN case only in that the B-BMIN case uses an arbitrary SCP. There are no differences between both from the point of view of cost, and so the O-BMIN case is not considered in order to perform the cost estimation. Once cost values are estimated, we identify which elements require an extra economic investment in building T-switches with current single-chip switches. In other words, we focused on identifying the expenses that any manufacturer, who would implement the T-switch approach, would need to cover apart from the expenses of implementing a traditional network like U-BMIN or L-BMIN.

TABLE II outlines the values of the cost metrics for the aforementioned three case studies we have considered. Next, some assumptions about these values have to be introduced for each case study:

1) U-BMIN
   This BMIN is built with $k \times k$ single-chip switches. We assume that:
   - The single-chip switches radix is $2k$.
   - The number of stages is n being $n = \log_k N$.
2) O-BMIN
   For this case, this BMIN is built with $k \times k$ T-switches. We assume that:
   - The internal single-chip switches radix is $k + \frac{k}{4} = \frac{5}{4}k$ being the switch arity $\frac{k}{2} + \frac{k}{8} = \frac{5}{8}k$. In Section IV we have proved that $\frac{k}{4}$ is the minimum number to avoid links interconnecting the internal switches become bottleneck under uniform traffic pattern.
   - The number of stages is also n.
3) L-BMIN
   Since this BMIN is built with $k' \times k'$ single-chip switches, we assume that:
   - The single-chip switches radix is $2k'$ where $k' < k$ is the arity.
   - There are n' stages being $n' = \log_{k'} N$.

In Fig. 7 we show the values of the cost metrics for our case studies varying the network size range. The results have been obtained considering $k = 16$ for the U-BMIN and O-BMIN cases, and $k' = 8$ for the L-BMIN case. The relation between k and k' has been chosen in order to keep the same network topology in all the cases. Fig. 7a plots the number of stages that are required in order to interconnect at least N terminals. In this sense, it is observed that the L-BMIN case needs more stages in comparison with the U-BMIN and O-BMIN cases.

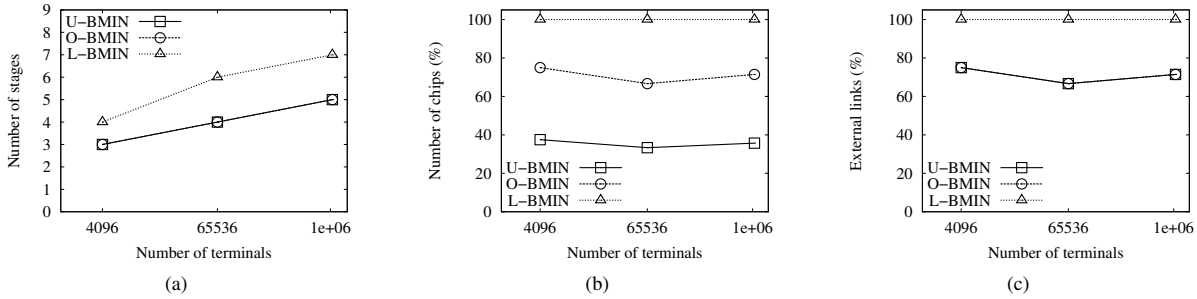


Fig. 7. Number of (a) stages, (b) chips, and (c) external links for our case studies. $k = 16$ for U-BMIN and O-BMIN, and $k' = 8$ for L-BMIN.

Fig. 7b shows the number of chips that are necessary for each case study. These numbers are represented as a percentage in respect of the number of chips used by L-BMIN (100%). The O-BMIN case (i.e., T-switches) uses at least 20% fewer chips than the L-BMIN case with independence of the number of terminals. Note that although the O-BMIN case uses the double of chips than the U-BMIN case, the chips that the U-BMIN case would require or they do not exist because the integration scale limitations, or if they are viable, are much more expensive because they are of a generation more recent than the smaller chips used in the O-BMIN case. Lastly, the number of external links are also represented as a percentage in respect of the L-BMIN case (100%) in Fig. 7c. In a similar way to the number of chips, the O-BMIN case needs 20% fewer external links than the L-BMIN case. This result is significant because the fewer external links (i.e., cables) the lower the network cost.

C. Cost Discussion

Resuming the cost estimation, we need to perform two comparisons to determine what is the extra cost that a O-BMIN implementation would require, and that cost would not be amortized by a future U-BMIN implementation when $k \times k$ single-chip switches would be available in the market.

The first comparison is between O-BMIN and L-BMIN:

- The O-BMIN case needs $C_L - C_O$ single-chip switches less than the L-BMIN case (Fig. 7b). The chip cost is determined by the design development cost and the silicon cost [11]. Since the chips needed to build T-switches are available in the market, no development cost would be required and the total switch cost would continue amortizing.
- The O-BMIN case requires $E_L - E_O$ external links less than the L-BMIN case (Fig. 7c). The number of global links are also decreased, and the length of cables must be shortened.
- Regarding the T-switch packaging, it is feasible since the two internal chips can be placed together in a motherboard with short links (printed in the board) interconnecting them.

Secondly, we also compare O-BMIN with U-BMIN:

- Although the O-BMIN case doubles the number of chips regarding the U-BMIN case (Fig. 7b) we should remember that $k \times k$ single-chip switches (used in U-BMIN) or are not available in the market due to integration technology limitations or they are much more expensive. Again, for the O-BMIN case the silicon cost is only necessary because the switch design has been already made.
- With independence of the number of network switches, the number of external links is the same in both cases $E_U = E_O = N \times n$ (Fig. 7c) so when the technology would evolve and $k \times k$ single-chip switches would be affordable, the links would remain valid and they would be reusable in future deployments. Thus, the cost per link is also reduced because their development cost is amortized.

Only in the O-BMIN case, a total of I_L links interconnecting the internal switches of T-switches are necessary. The cost of an internal links is several orders of magnitude lower than an external link cost.

Summing up, we estimate that the O-BMIN cost is due to the implementation of two single-chip switches in one motherboard. The cost of implementing links interconnecting the internal switches is included in the motherboard cost. Moreover, the cabling cost of the O-BMIN case does not mean an extra cost with regard to the U-BMIN case. In our best knowledge, the Sun Blade 6048 NEM is based on a T-switch approach (Section VII).

VII. Related Work

In this section we review of existing proposals in the literature of high-radix switches, which are mainly focused on solving the scaling problems from traditional switch designs.

The YARC proposal is the high-radix switch used by the Cray BlackWidow [14]. Its internal organization is defined in [15] and is based on increasing the number of ports considering more thin channels instead of less wide channels. Traditional switch designs with fewer ports cannot be adapted to high-radix switches, because the traditional centralized organization does not scale properly.

On the other hand, the Partitioned Crossbar Input Queued [16] is a more recent proposal for the internal organization of high-radix switches, and deals with one of the main constraints in high-radix switch design, the excessive memory requirements. The authors replace the central crossbar by several internal crossbars improving the readability of the

buffers without increasing hardware cost. Moreover, the switch implements an efficient congestion control mechanism, thus, completely eliminating HOL blocking [17] at both switch and network levels.

Regarding the alternative for building high-radix switches using low-radix switches, the Oracles's Sun Blade 6048 Infiniband QDR Switched Network Express Module (NEM) [18] has already implemented this strategy, offering the ability to connect up to 12 dual-node blades in a single shelf. Each NEM provides 12 connections from each of the 36-port Mellanox's InfiniScale IV switches. A total of 24 connections are used to communicate with the two compute nodes on each of the dual-node server blades, with 9 ports used to connect the two switches together. The 30 remaining ports (15 per switch chip) are used as links to either other NEMs, or to external switches.

On the other hand, the Dragonfly topology [19] uses a group of routers as a virtual router to increase the effective radix of the network. However, no formal analysis have been provided.

To the best of our knowledge, there are currently no published formal studies for obtaining the optimal switch-level connection pattern of high-radix switches composed of several internal single-chip switches interconnected each other.

VIII. Conclusions

In this paper we have described an interesting alternative for building high-radix switches. The idea consists in combining several low-radix switches to obtain a switch with a greater number of ports. Although seemingly simple, this strategy raises key design challenges in order to these high-radix switches achieve the best performance. In this sense, we have discussed about important aspects related to the internal structure of this kind of switches.

The ports offered by these high-radix switches are obtained from the ports of the smaller switches from which high-radix switches are built. Internal connections between both groups of ports determine a connection pattern, which becomes crucial in the high-radix behavior. On the other hand, it is also essential to determine the most adequate subnetwork interconnecting the internal switches, in order to avoid these connections become a bottleneck.

To check the potential of this alternative, we have performed a formal analysis and evaluated under certain conditions the performance of networks based on combined switches composed of two smaller identical switches. The results are similar to those obtained with the same networks, but using single-chip switches with the same number of ports as combined switches. Moreover, results highlight the importance of selecting the correct number of ports of the internal switches for interconnecting them.

Finally, we have estimated the implementation cost of this alternative that effectively increases 60% the single-chip switch radix while the networks using this alternative provide the same performance as the traditional networks with higher single-chip switch radix. It should be noted that those single-chip switches with higher radix are not available in the market due to integration scale limitations, or if they are viable, are much more expensive because they are of a generation more recent than the smaller chips used in the O-BMIN case. Consequently, we have discussed about the cost of this alternative for building combined switches, and we have concluded that the most relevant cost is located in the implementation of the motherboard that hosts the two internal switches. Other expenses are not specific of this alternative, and they are amortized by traditional network implementations.

References

[1] J. J. Dongarra, H. W. Meuer, and E. Strohmaier, "TOP500 Supercomputer Sites," Nov. 2010.

[2] H. Wang, L.-S. Peh, and S. Malik, "Power-driven design of router microarchitectures in on-chip networks," in *Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, ser. MICRO 36. Washington, DC, USA: IEEE Computer Society, 2003.

[3] M. Gusat, F. Abel, F. Gramsamer, R. Luijten, C. Minkenberg, and M. Verhappen, "Stability degree of switches with finite buffers and non-negligible round-trip time," *International Conference on Computer, Communication and Networking*, vol. 27, no. 5–6, 2003.

[4] C. Minkenberg, F. Abel, P. Muller, R. Krishnamurthy, M. Gusat, and B. R. Hemenway, "Control path implementation for a low-latency optical HPC switch," in *Proceedings of the 13th Symposium on High Performance Interconnects*, ser. HOTI'05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 29–35.

[5] C. Minkenberg and M. Gusat, "Speculative flow control for high-radix datacenter interconnect routers," *Parallel and Distributed Processing Symposium, International*, vol. 0, pp. 1–10, 2007.

[6] "International Technology Roadmap for Semiconductors: 2010 Update," 2010, www.itrs.net/Links/2010ITRS/Home2010.htm.

[7] W. Dally, "Virtual-channel flow control," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 3, no. 2, pp. 194 –205, mar 1992.

[8] J. A. Villar, F. J. Andújar, F. J. Alfaro, J. L. Sánchez, and J. Duato, "An alternative for building high-radix switches: Formalization and configuration methodology," Department of Computing Systems. University of Castilla-La Mancha, Tech. Rep. DIAB-11-02-1, 2011, www.dsi.uclm.es/descargas/thecnicalreports/DIAB-11-02-1/diab-11-02-1.pdf.

[9] F. Petrini and M. Vanneschi, "K-ary n-trees: High performance networks for massively parallel architectures," Tech. Rep., 1995.

[10] C. Gómez, F. Gilabert, M. E. Gómez, P. Lopez, and J. Duato, "Deterministic versus adaptive routing in fat-trees." Los Alamitos, CA, USA: IEEE Computer Society, 2007.

[11] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.

[12] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection networks. An engineering approach*. Morgan Kaufmann Publishers Inc., 2003.

[13] M. T. Inc., "Infiniscale III 3rd generation infiniband switch architecture," www.mellanox.com/related-docs/prod_silicon/PB_InfiniScale_III.pdf.

[14] S. Scott, D. Abts, J. Kim, and W. J. Dally, "The BlackWidow high-radix Clos network," *SIGARCH Comput. Archit. News*, vol. 34, no. 2, pp. 16–28, 2006.

[15] J. Kim, W. J. Dally, B. Towles, and A. K. Gupta, "Microarchitecture of a high-radix router," *SIGARCH Comput. Archit. News*, vol. 33, no. 2, pp. 420–431, 2005.

[16] G. Mora, J. Flich, J. Duato, P. López, E. Baydal, and O. Lysne, "Towards an efficient switch architecture for high-radix switches," in *ANCS '06: Proceedings of the 2006 ACM/IEEE symposium on Architecture for networking and communications systems*. New York, NY, USA: ACM, 2006, pp. 11–20.

[17] N. McKeown, "The islip scheduling algorithm for input-queued switches," *Networking, IEEE/ACM Transactions on*, vol. 7, no. 2, pp. 188 –201, apr 1999.

[18] "Sun datacenter Infiniband switch 36, Sun datacenter Infiniband switch 72, Sun datacenter Infiniband switch 648: Architecture and deployment," April 2010.

[19] J. Kim, W. J. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable Dragonfly topology," in *ISCA '08: Proceedings of the 35th Annual International Symposium on Computer Architecture*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 77–88.