Leveraging Partially Faulty Links Usage for Enhancing Yield and Performance in Networks-on-Chip

Maurizio Palesi, Member, IEEE, Shashi Kumar, Senior Member, IEEE, and Vincenzo Catania

Abstract—The communication infrastructure of a complex multicore system-on-a-chip is getting an increasing fraction of the overall chip area. According to the International Technology Roadmap for Semiconductors, killer defect density does not decrease over successive technology generations. For this reason, the probability that a manufacturing defect affects the communication system is predicted to increase. In this paper, we deal with manufacturing defects which affect the links in a network-on-chip-based interconnection system. The goal of this paper is to show that by using effective routing functions, supported by appropriate selection policies and with a limited amount of extra logic in the router, it is easy to exploit partially faulty links to improve the performance of the system. We show that, instead of discarding partially faulty links, they can be used at reduced capacity to improve the distribution of the traffic over the network, yielding performance and power improvements. We couple an application-specific routing function with a set of selection policies which are aware of link fault distribution and evaluate them on both synthetic traffic and a real complex multimedia application. We also present an implementation of the router, augmented with the extra logic, to support both the proposed selection functions and the transmission of messages over partially faulty links. We analyze the router in terms of silicon area, timing, and power dissipation.

Index Terms—Application-specific routing, congestion, fault tolerance, network-on-chip, performance analysis, router design, routing algorithm.

I. INTRODUCTION

THE SEMICONDUCTOR industry operates in an environment of exponentially decreasing product prices, which put semiconductor manufacturers under time-to-market pressure. Profitability is derived from an early and successful yield ramp. The sooner a semiconductor manufacturer generates high yield, the earlier the manufacturer ramps to volume production, and the more profitable the semiconductor manufacturer’s integrated circuit venture is likely to be [1].

The complexity which characterizes the current and future generations of system-on-a-chip (SoC) is leading toward the definition of new methodologies and techniques aimed at improving the manufacturing yield. As chips become larger and feature size shrinks, some parts of the chip will more and more frequently be affected by manufacturing defects.

In this case, two manufacturing scenarios can be envisaged. A simple approach is to discard the chips affected by manufacturing defects. An alternative solution is that in which faulty blocks of the chip are disabled or isolated, thus obtaining a “depowered” chip that will be offered at the lower end of the market. Of course, this solution requires the design to be augmented with mechanisms to disable and isolate blocks of the system during post manufacturing. There are, for instance, versions of the UltraSparc T1 microprocessor with four and six cores obtained from the original eight-core version when one to four cores are affected by manufacturing defects [2]. Of course, this solution requires additional hardware support which allows the faulty blocks of the chip to be isolated and it can be used only in particular circumstances (i.e., the faults affect blocks whose functionalities can be carried out by another fault-free block).

The importance of interconnects on a chip has outrun the importance of transistors as a dominant factor of performance, power, and reliability [3]-[5]. Due to the fact that the effective chip area can increase or decrease with each successive technology generation, killer defect density does not always decrease over time. As feature sizes shrink and system complexity increases, faults affecting the communication infrastructure of an SoC are predicted to become more and more important [6]. One of the main reasons for this trend is that, due to the increasingly important role played by the interconnection system, it occupies quite a high percentage of the overall chip area (e.g., over 30% in the Intel TeraScale chip).

Sometimes, a manufacturing defect which affects the communication system is not so destructive. That is, depending on the gravity of the defect, communications can still be performed although it will probably exhibit lower performance (higher delay, lower throughput, so on). By way of example, let us consider a transmitter (TX) and a receiver (RX) which communicate by means of an n-bit link. Let us assume that, due to a manufacturing defect, m of the n lines of the link...
are faulty. The partially faulty link could still be used if both the transmitter and the receiver take care not to use the faulty link lines to communicate. Of course, this will require additional logic to split and reassemble data at the interfaces of the transmitter and the receiver modules (see Fig. 1), and it will negatively impact the communication performance. However, the chip can still be used (e.g., it can be shipped at a reduced cost and target the lower end of the market).

The network-on-chip (NoC) paradigm is emerging as the best implementation option for building single-chip multicore embedded systems. The probability of having partially faulty links might considerably increase in future (e.g., many-core system) architectures, which are expected to be implemented with a 22 nm technology by 2015 [7]. This problem becomes quite significant in NoC systems due to the large number of wires that form a link and the high number of links in the whole chip [8]. Here, we focus on faults affecting NoC links as they are longer and span more metal layers than the other wires in a chip. If an NoC is expected to take around 20% of the total chip area/wires, then it is not worth addressing NoC link faults since the improvement will be marginal. However, that 20% of NoC links will be wires that are difficult to manufacture—long and needing many vias—so the defect density for these wires will be higher.

In this paper, we show that it is often better to use partially faulty links in an NoC instead of discarding them during the design of the routing function. This, of course, requires NoC routers to be augmented with logic implementing new functionalities (like flit splitting and split reassembling) and new selection policies that dynamically determine when a certain partially faulty link should be used and when it should be avoided for communication. More specifically, in this paper we present five different strategies aimed at improving both the performance and the energy consumption of NoC-based SoCs in which some network links are partially faulty. The proposed strategies are analyzed with different fault combinations and in several traffic scenarios.

The instantiation of the new functionalities into the router to manage transfers on partially faulty links and the new selection policies have an impact on the area, timing, and power of the router. The impact is carefully analyzed in this paper to assess whether the overhead introduced nullifies the improvements achieved.

II. RELATED WORK

As technology scales toward deep submicrometer the manufacturing defects and variations become prominent [9]. In the context of reliable and fault-tolerant on-chip communication, many approaches have been presented in the literature to tackle the problem of dynamic error control due to the increased impact on signal reliability of noise sources, such as crosstalk, power-supply noise, electromagnetic interference, and soft errors [10], [11]. In this paper, the attention is focused on extrinsic manufacturing defects which affect the links of an NoC. A representative set of work in this context is briefly introduced as follows.

In order to improve the yield and reliability of multicore SoCs, Greco et al. present a self-repair method for the interconnect fabrics of integrated multicore systems [13]. The method is based on the use of redundant links and crosspoints, and improves both post-manufacturing yield and life-time reliability of on-chip communication fabrics. If a link contains some faulty wires, the basic idea is to choose a complete set of nonfaulty wires using a crossbar to perform the transmission. Link structures for NoC that have properties for tolerating efficiently transient, intermittent, and permanent errors have been proposed by Lehtonen et al. [14]. This paper introduces two approaches for tackling the intermittent and permanent errors. In the first approach, spare wires are introduced together with reconfiguration circuitry. The other approach uses time redundancy, the transmission is split into two parts, where the data is doubled.

Tamhankar et al. present a timing-error-tolerant design method to design the individual components of the NoC (switches, links, and network interfaces) so that the communication subsystem can be clocked at a much higher frequency than a traditional conservative design [15]. The NoC is designed to tolerate timing errors (both transient and permanent) caused by interferences, coupling, noise, or variability. Hernandez et al. present and evaluate several alternatives to mitigate the effect of variability that results in delay variations and, therefore, maximum operating frequency of each wire of a link in an NoC [8]. The analysis reported in this paper is limited to the variation of maximum throughput that can be achieved using three different strategies, namely frequency reduction, split transmission, and discarded wires. However, the overhead of the additional logic to be implemented in the router to support some of the proposed mechanisms and its impact on cost (silicon area), power, and critical path is not discussed in this paper. A first attempt to assess the feasibility of adding new functionalities to the router to support the use of partially faulty links for flow forwarding is discussed in our previous work [16].

The above work addresses the issues of process variability and manufacturing defects affecting the interconnection system as far as the architectural viewpoint. However, to be effective, such architectural innovations must be supported by appropriate routing algorithms. For instance, without an appropriate routing algorithm, it is expected that traffic congestion will appear close to the regions where faults are localized. Hot-spot traffic congestion has a very negative impact on overall system performance [17]. Thus, the need...
of mechanisms able to balance the traffic over the network is envisi-able. In this direction, some researchers [18], [19] have proposed on-line techniques to reduce this problem. On-line measurement, estimation, and control of traffic is very costly and is not very effective if traffic variance is high. Earlier in [20] we had proposed a technique to estimate and predict such congestion situations. In [20], we also proposed a method to use available adaptability in the routing algorithm for link load balancing, especially when using application-specific routing algorithms. In this paper, we build on our previous work presented in [20] and [21]. We propose and compare many routing algorithms which combine the information regarding communication behavior of the mapped and scheduled application (which pairs of cores in NoC communicate and which pairs never communicate) along with capacity of various links, including available capacity of partially faulty links to achieve high routing performance and balanced link loads.

III. ANALYSIS OF THE ACHIEVABLE LATENCY IMPROVEMENT

Before we present the detail of the proposed approach, it is useful to perform a quantitative analysis [22] which allows to estimate the expected performance improvements that can be achieved when the partially faulty links are exposed to the routing function.

A. Quantitative Analysis at Zero Load

In this analysis, we refer neither to a particular selection policy nor to a particular routing function. That is, although the use of partially faulty links increases the degree of freedom in the process of selecting a routing path, the analysis we perform in this section does not exploit any load balancing technique aimed at optimizing the distribution of the traffic over the network. Conversely, in the following quantitative analysis we emphasize the performance improvements (measured in terms of flit latency) that can be achieved thanks to the fact that the use of partially faulty links could make it possible to reach the destination in less hops. For this reason the achievable performance results that will be discussed in this section represent just a lower bound of the performance improvements that can be achieved when the use of partially faulty links is coupled with an effective routing function and an appropriate selection policy.

The latency for a flit traveling on a path of \( h \) hops is

\[
T = T_r(h + 1) + h T_I,
\]

where \( T_r \) is latency of a router (i.e., the time required for a flit to traverse a router) and \( T_I \) is the link latency (i.e., the time required for a flit to traverse a link). Similarly, the latency for a flit traveling on a path of \( h' \) hops in which \( h'_r \) links are fault free and \( h'_I \) links are partially faulty is

\[
T' = T'_r(h' + 1) + h'_r T'_r + h'_I T'_I
\]

where \( T'_r \) is the latency of the router augmented with the logic to manage transmission on partially faulty links, and \( T'_I \) is the time required for a flit to traverse a partially faulty link. Thus, the use of routing paths containing partially faulty links is convenient, in terms of delay, if and only if \( T' < T \). In this case, the speedup of not discarding partially faulty links when routing paths are built is

\[
\text{Speedup} = \frac{T}{T'} = \frac{T_r(h + 1) + h T_I}{T'_r(h' + 1) + h'_r T'_r + h'_I T'_I}
\]

Let us now manipulate (1) to express it in terms of relative quantities. We define the following relative parameters:

1. \( \alpha \approx T_r/T_r' \), is the delay slowdown factor of the augmented router which supports transfers through partially faulty links over the base router;
2. \( \beta \approx T_I/T_I' \), is the slowdown transmission time factor to transmit a flit through a partially faulty link over transmission through a fault free link;
3. \( \gamma \approx h'/h \), it is the hop count reduction factor, that is the path length reduction when passing from the case in which partially faulty links are discarded to the case in which partially faulty links are used;
4. \( \delta \approx T_I/T_r \), it is the link to router latency ratio;
5. \( \epsilon \approx h'_r/h'_I \), for a routing path obtained without discarding partially faulty links, it represents the ratio between the number of fault free links and the number of partially faulty links.

Using the above definitions, and performing some symbolic algebraic manipulations on (1) we obtain

\[
\text{Speedup} = \frac{1 + h \delta + 1}{\epsilon (1 + \gamma) + (\beta + \gamma) + \alpha}
\]

Equation (2) expresses the improvement in delay of sending a flit from a source node to a destination node when partially faulty links are used as compared to when only fault free links are used in function of the relative parameters \( \alpha, \beta, \gamma, \delta, \) and \( \epsilon \).

To get some confidence with the speedup values that can be achieved in practical cases, let us consider as baseline a routing path of ten hops, that is \( h = 10 \). If we consider a three-stage router pipeline, that is, \( T_r = 3 T_I \), which means \( \delta = 1/3 \). Let us suppose that the flit transmission time through partially faulty links is twice that required to transmit a flit through a fault free link. That is, \( T_I = 2 T_r \), which means \( \beta = 2 \). Finally, let us assume that on average a routing path computed without discarding partially faulty links contains a number of fault free links which is one order of magnitude greater than the number of partially faulty links in the path. That is, \( h'_r = 10 h'_I \), which means \( \epsilon = 1/10 \).

Fig. 2(a) shows the contour plot of communication speedup using \( \alpha \) and \( \gamma \) as free parameters. We analyse the design space spanned by routers able to manage partially faulty links which are up to 1.5 times slower than the baseline router (i.e., \( 1 \leq \alpha \leq 1.5 \)) and routing functions that, by exploiting partially faulty links, make it possible to reduce the path length up to a factor 2 (i.e., \( 0.5 \leq \gamma \leq 1 \)). As can be observed the use of partially faulty links is useful (i.e., speedup > 1) for routers which are up to 1.5 times slower than the baseline router if the exploitation of partial faulty links results in path length reduction greater than 25%. However, as will be seen in Section VIII, the introduction of the logic implementing the mechanism which allow to exploit partially faulty links...
does not impact the critical path of a router if we consider a pipeline implementation. In this case, \( \alpha = 1 \) and the use of partially faulty links is convenient if path length reduction is at least 5%.

The impact of path length is analysed in Fig. 2(b) which shows the contour plot of the speedup using the baseline path length, \( h \), and the hop count reduction factor, \( \gamma \), as free parameters. Here we consider \( \alpha = 1 \) and \( \varepsilon = 10 \). As can be observed, the speedup is quite insensitive to \( h \). For instance, for a path length reduction of 25% (\( \gamma = 0.75 \)) the speedup remains confined in the band \([1.2, 1.3]\) when \( h \) increases from two to 16 hops.

Finally, Fig. 2(c) shows the impact on speedup due to the different distributions between the number of partially faulty links and the number of fault free link, \( \varepsilon \), for different baseline path length, \( h \). Here we consider \( \gamma = 0.5 \). As can be observed, a certain speedup value obtained for a given path length can be obtained for a shorter path length but at a much higher \( \varepsilon \) value.

It should be pointed out that the above analysis refers to the case in which all the admissible minimal routing paths contain at least a partially faulty link. In fact, only in this case there will be a difference in routing path length between the two scenarios. To obtain the effective speedup, we need to multiply (1) by the probability of a communication using a faulty link.

### B. Qualitative Analysis at High Load

The analysis in the previous subsection demonstrates that using the partially faulty links with reduced capacity has a potential of reducing packet latency in general. The advantage of using partially faulty links grows dramatically at higher communication traffic. Ignoring a partially faulty link will lead to diversion of traffic, which would have used this link if it was not faulty, to the neighboring links. This will cause these neighboring links to get highly congested at high load. As we know, the average latency of a packet increases exponentially at high load (Fig. 3). The overall latency of the network will be determined by the latencies of communications using the congested links. Use of partially faulty links will relieve the pressure on these congested links and will move the load-latency point to the left as shown in Fig. 3. Due to super-linear nature of the curve at high load values, a small reduction in load (congestion) leads to big reduction in latency.

### IV. TERMINOLOGY AND PROBLEM FORMULATION

A link can be faulty with several levels of severity. A \( \alpha \% \) faulty link is a link in which the capacity of the partially faulty link is reduced by \( \alpha \) percent with respect to the normal link capacity.

We model the network topology by means of a topology graph. The topology graph, \( TG = G(N, L) \), is a directed graph where \( N \) is the set of network nodes, and \( L \) is the set of network links. Link \( l_{ij} = (n_i, n_j) \) connects node \( n_i \in N \) to node \( n_j \in N \). Given a link \( l \in L \), the function \( FD(l) \) returns the fault degree of \( l \).

Faults may result in change of the original network topology. In this paper, we focus on links faults. We indicate with \( TG(f) \) a filtered topology graph. It is a subgraph of the \( TG \) obtained by removing all the links with a fault degree greater than \( f \). Formally, \( TG(f) = G(N, L(f)) \), where \( L(f) = \{ l \in L : FD(l) \leq f \} \).
A router can be schematized as a cascade of two main blocks, implementing a routing function and a selection function (see Fig. 4). A routing function, for a given input link and a given destination node, returns a set of admissible output links where the message can be forwarded. A selection function selects one link from the set of admissible links for forwarding the incoming packet. The selection function uses information regarding traffic conditions in the network for the decision.

Let us consider an SoC whose NoC-based interconnection system is affected by some manufacturing defects. We classify such defects (or failures) as core failure, network interface failure, link failure, and router failure. Although in this paper we focus on link failures, it should be pointed out that the other kind of faults can be simply mapped to link faults. For instance, a router fault can be modeled as 100% faults in all its input and output links. If a link is partially faulty (i.e., not all the bit-lines that form the link are faulty) two general solutions are: 1) the entire link is considered faulty and not used for any transfer; and 2) only the "healthy" lines are used for transfers.

The first solution is the most straightforward and simple to implement. It does not require additional hardware support, but it only needs that the routing function is computed assuming a network topology in which faulty links are not present. The second solution requires additional hardware support for flit splitting and flit reassembling (TFM_TX and TFM_RX blocks in Fig. 5) when a partially faulty link is used.

For practical reasons, to simplify the design of a router to handle partially faulty links, we assume that faults result in reduction of link capacity by 25%, 50%, 75%, or 100%. Although this second solution adds complexity to the router design, in this paper we show that the performance improvement obtained, especially at high traffic loads, justifies such overhead.

Therefore, problem we are solving in the paper can be stated as follows. Given a filtered network topology and information about communication requirements of the application, we are to design routing functions and selection functions for various routers such that: 1) all packets use minimal routes; and 2) traffic load on links is balanced as much as possible. We borrow the solution for the design of routing function from application-specific routing algorithm (APSRA) methodology [21]. The specific contributions of this paper are: 1) the design of a router which can handle partially faulty links; and 2) a set of selection functions which distribute load uniformly over the network and avoid congestion.

V. OVERVIEW OF THE SOLUTION

The basic idea behind the proposed approach can be formulated by means of a practical real life example. Let us suppose we have to reach a certain destination by car. If several routing alternatives are available, we usually select one of them based on some criteria like traffic condition, roads conditions, and so on. For instance, in low traffic condition we will use a route with good roads (wide, reliable, and so on). On the other hand, in heavy traffic condition, we will be more flexible to choose a route in which some roads, although not in perfect conditions, are less congested.

Fig. 6 shows the general design flow for the design of application-specific routing algorithms. The block bounded with dashed lines in the diagram represents our contribution. The inputs of the design flow are the application (represented by means of a set of concurrent task graphs), and the network topology. The first step consists of removing from the topology graph all those edges whose fault degree exceeds a user defined threshold. The filtered topology graph along with the application specification is the inputs of APSRA methodology [21]. The output is a set of routing tables which define a highly adaptive, deadlock-free, and application-specific routing function. The routing function, along with the filtered topology graph and the application are inputs for a load balancing techniques whose objective is to determine the best set of selection probabilities which result in a uniform distribution of the traffic over the network. Such selection probabilities are
stored as additional information in the routing tables. Finally, the extended routing tables are used to configure the NoC.

VI. STRATEGIES WITHOUT LOAD BALANCING

In this section, we present the following three strategies aimed at managing partially faulty networks:

1) faulty links elimination strategy (FE);
2) partially faulty links usage strategy (FU);
3) partially faulty links usage with look-ahead strategy (FUL).

Each strategy is discussed in detail in the next subsections.

A. Faulty Links Elimination Strategy

In the FE, the routing function is computed on the basis of the network filtered by all the fully faulty or partially faulty links. That is, the routing paths are determined considering TG(0).

If we indicate with $L_{fa} = \{l_1, l_2, \ldots, l_n\}$ the set of admissible output links returned by the routing function in a given node, for a given input, and a given destination, the selection function $SE$ selects any one of them randomly [(a) in Table I].

B. Partially Faulty Links Usage Strategy

In the partially FU, the routing function is computed on the basis of the network filtered by all partially faulty links with a fault degree greater than a user defined faulty threshold, $T$. That is, the routing paths are determined considering TG($T$).

The selection function is mainly based on the ability of discriminating between low and high traffic conditions. It is assumed that each router is equipped with saturating counters (one for each output link) which are used as follows. When a flit is transmitted toward a certain output link, the counter associated with that output link is incremented if the number of free available slots in the receiving first in, first out (FIFO) buffer is below a given threshold $T_i$. If the condition is not satisfied, the counter is decremented. The link is considered being in high traffic condition if the value in its associated counter is greater than a given threshold $T_i$. Thus, the selection function discriminates between low and high traffic condition by means of a majority voting policy between the estimated traffic condition status of the admissible output links returned by the routing function.

Overall, the selection function works as follows. In low traffic conditions, the best link in $L_{fa}$ (i.e., the link with the lowest fault degree) is always chosen. In high traffic conditions, a link $l_i \in L_{fa}$ is selected with a probability $Pr^{H/L}(l_i)$ which is inversely proportional to its fault degree. Formally, the selection probability, $SE_{FU}$, is defined as shown in (b) in Table I. It should be pointed out that the set of probabilities, $Pr^{H/L}$, is computed off-line as

$$Pr^{H/L}(l_i) = \frac{1 - FD_l(l_i)}{\sum_{i=1}^{n}(1 - FD_l(l_j))}$$

and stored in the routing table.

C. Partially Faulty Links Usage With Look-Ahead Strategy

Like FU strategy, in the partially FUL, the routing function is computed on the basis of TG($T$), where $T$ is a user defined fault degree threshold. The selection function used in FU does not take into consideration the entire path, but, instead, it takes the decision based solely on the quality of the next link. This can cause inefficiencies when, although the next link is of high-quality (e.g., fault free), the rest of the path(s), in which the message will be obliged to travel on is/are formed by many low-quality links.

This problem can be partially solved as follows. For each destination $n_d \in N$, each router holds a word of $n$ bits ($n$ is the number of output links) called ERM (equivalent resistance mask). The $i$th bit of ERM is set if the equivalent resistance of paths having as first link the output link $l_i$ and ending at node $n_d$ is minimum. The equivalent resistance, $ER$, of a set of paths with the same starting node and the same ending node (re-convergent set of paths) is computed applying Kirchhoff’s laws. Network nodes are considered as circuit nodes and network links are considered as electrical resistances whose value is the fault degree of that link. Formally, if $\bar{P}$ is a set of re-convergent paths, the equivalent resistance of $\bar{P}$ is

$$ER(\bar{P}) = \prod_{l_p \in \bar{P}} R_p(l_p)$$

where $R_p(l_p)$ is the resistance of a path $P$ computed as

$$R_p(l_p) = \sum_{l_{ij} \in l_p} FD(l_{ij})$$

Of course, the ERM's are computed off-line and stored in the routing table.

Fig. 7(a) shows an example of how the ERM is computed. In Fig. 7(a), the three paths allowed by the routing function from a source node $n_s$ to a destination node $n_d$ are $\bar{P}_1$. The $i$th bit of ERM is set if the path starting from $l_i$ to $n_d$, which is $p_i$, is

$$ER(p_i) = R_p(p_i) = 0.$$ 

The re-convergent paths starting from $l_i$ are $P_2$ and $P_3$

$$ER(\{p_2, p_3\}) = \sum_{l_{ij} \in l_p} FD_l(l_{ij}) = 0.$$ 

As $ER(p_i) < ER(\{p_2, p_3\})$, thus $ERM = \{1, 0\}$. Overall, the selection function works as follows. In low traffic load conditions, one of the admissible output links $l_i$, such that the $i$th bit of ERM is set, is randomly chosen. Note that, at least one bit of ERM must be set. In addition, if $l_i$ is such...
that there is a fault-free path from \( l \) to the destination node, then the \( i \)th bit of \( ERM \) is set. This means that, in low traffic conditions, if multiple fault-free paths exist, they are always used. In high traffic conditions, the same scheme as discussed in FU strategy is used. Formally, the selection probability used in FUL, \( S_{FU} \), is defined as shown in (c) in Table I.

### D. Other Strategies

Several other selection strategies can be defined. For instance, between FE and FU one can think of a strategy in which partially faulty links are used based on a random selection function which works as follow. In low traffic conditions, the selection function randomly selects a fault-free output link belonging to the set of admissible output links \( L_{ao} \). In high traffic conditions, the selection function randomly selects an output link belonging to \( L_{ao} \). That is, in low traffic conditions the selection is restricted to the subset of admissible output links which are fault free, whereas in high traffic conditions, every admissible output link (both fault-free and partially faulty) can be selected with the same probability. Formally, such kind of selection function is defined as

\[
S(l) = \begin{cases} 
1 & \text{if } l \in L_{ao} \\
0 & \text{if } l \notin L_{ao} 
\end{cases} 
\]

where \( F(L_{ao}) \) returns the set of partially faulty output links belonging to \( L_{ao} \).

In this paper, we do not report the performance and cost results obtained for this strategy since we found that it is dominated by FE both in terms of performance and cost. With regard to performance, we found that the average latency obtained with FE is very close and in some case better than that obtained by (3) (at least for the traffic patterns we considered in this paper). In terms of cost, to implement (3), the routing table must be augmented to store the information about the position of partially faulty admissible output links. As we will see in Section VII, the routing table represents one of the main blocks which determines the cost of the router. Thus, an increase of the routing table which is not amortized by a substantial improvement in performance represents a bad design choice.

### VII. Strategies With Load Balancing

In the previous section, we have described three possible strategies aimed at managing partially faulty links. One of them (FE) is very straightforward and forces the routing algorithm not to use partially faulty links. The other two, FU and FUL, allow the routing function to generate a set of admissible outputs, containing some partially faulty links but define a selection function in such a way that partially faulty links are selected only when it is convenient to use them. In this section, we improve the selection function by using an accurate off-line traffic analysis to determine the optimal set of selection probabilities which allow a uniform distribution of the traffic over the network. Based on this, we obtain the following three variants of the previously discussed strategies:

1. Faulty links elimination with load balancing strategy (FE+LB);
2. Partially faulty links usage with load balancing strategy (FU+LB);

Before describing these new strategies, let us present the technique used to determine the optimal set of selection probabilities.

### A. Load Balancing Technique

For a given routing function, and a given communication pattern, in this subsection we formulate and solve the problem...
of determining the optimal set of selection probabilities which allows uniform distribution of the traffic over the network. In order to accomplish this goal, the variance of the traffic function (TF) must be minimized. Given a topology graph $TG = G(N, L)$, for any network link $l \in L$, $TF(l, X)$ returns an estimation of the traffic load on link $l$ when the set of selection probabilities $X$ is used. Such estimation depends on the selection probabilities as follows:

$$TF(l, X) = \sum_{c \in X} V(c) Pr^{(X)}(c, l)$$

where $C$ is the set of communications (i.e., the set of communicating node pairs), $V(c)$ is the traffic volume of communication $c$, and $Pr^{(X)}(c, l)$ is the probability that communication $c$ uses link $l$:

$$Pr^{(X)}(c, P) = \prod_{l \in X(c, P)} \prod_{i \in C_{l}} x_{i}$$

where $X(c, P)$ is the set of routing paths containing link $l$ which can be used by communication $c$. $Pr^{(X)}(c, P)$ is the probability that a communication uses the links in the sub-path of $P$ up to link $l$. Let us indicate with $[l_i, l_2, \ldots, l_n = l, \ldots]$ the succession of links which forms $P$. Then

$$Pr^{(X)}(c, P) = \prod_{l \in X(c, P)} \prod_{i \in C_{l}} x_{i}$$

where $x_{i} \in X$ is the selection probability that a flit destined to $d$ and received from input link $l$, at node $n$ will be forwarded to output link $L$. Please note that, as both $l \in L$ and $c \in C$ are node pairs, we indicated with $l_{i}$ and $c_{i}$ the source node of link $l$ and the destination node of communication $c$, respectively.

To obtain a uniform distribution of the traffic over the entire network, we need to determine the set of selection probabilities $X$ in such a way as to minimize the variance of $TF$. Formally, we solve the following optimization problem:

$$\min_{X} \text{var}([TF(l, X)]) = \min_{X} \sum_{l \in L} \left\{ TF(l, X) - \text{mean}[TF(l, X)] \right\}^2$$

such that the following linear constraints are satisfied:

$$0 \leq x_{i} \leq 1 \quad \forall \ x_{i} \in X \quad (5)$$

$$\sum_{l \in L} x_{i} = 1 \quad \forall \ n, d \in N \quad \forall \ l_{i} \in L. \quad (6)$$

Constraint (5) assures that all selection probabilities are in the range $[0, 1]$. Constraint (6) assures that the sum of the selection probabilities for a given node, input link, and destination is 1.

The above problem is very complex for several reasons. It is a constrained optimization problem and the function to be minimized is nonlinear. In addition, the number of decision variables (selection probabilities) is very large. For instance, considering a network where routers have $N_i$ input ports and $N_o$ output ports, an upper bound on the number of decision variables is $|N| \times N_i \times (N_o - 1) \times (|N| - 1)$.

The $-1$ in the third factor is due to the fact that for a given input port and for a given destination one of the $N_s$ selection probabilities can be computed by difference between 1 and the sum of the rest of probabilities. The $-1$ in the last factor is because a node cannot be a destination to itself.

We use the sequential quadratic programming (SQP) optimization technique [23] to solve (4) satisfying constraints (5) and (6). The SQP algorithm has been one of the most successful general methods for solving large-scale nonlinear constrained optimization problems. The SQP algorithm is a generalization of Newton’s method for unconstrained optimization in that it finds a step away from the current point by minimizing a quadratic model of the problem. The main characteristic of SQP is that it attempts to solve a nonlinear problem directly rather than convert it to a sequence of unconstrained minimization problems.

### B. Load Balancing-Based Strategies

FE+LB, FU+LB, and FUL+LB strategies, respectively, descend from FE, FU, and FUL. They work similar to their respective parents, except that in low traffic conditions selection probabilities computed by means of the load balancing strategy are used in place of the $Pr^{(F)}$. In high traffic conditions, $Pr^{(F)}(l, i, l_i, l_o)$ are replaced with the following selection probabilities:

$$Pr^{(LB)}(l, i, l_i, l_o) = P_{LB}(l, i, l_i, l_o) \sqrt{Pr^{(F)}(l, i, l_i, l_o)}$$

where $Pr^{(LB)}(l, i, l_i, l_o)$ is the selection probability of output link $l$, computed by means of the load balancing strategy (LB)\(^1\). The rationale behind the definition of these selection probabilities is that from the experimental analysis we observed that best results are obtained when more weight is assigned to $Pr^{(LB)}$ than to $Pr^{(F)}$. This means that the latter probability should tend to zero more slowly than the first one. The square root function is an appropriate way to mimic such behavior.

Once again, such probabilities are computed off-line and stored in the routing table. The selection functions used in LB-based strategies are defined in (d), (e), and (f) in Table I.

### VIII. Implications on Router Design

In this section, we analyze the impact on area, timing, and power consumption due to the additional hardware required to support the selection strategies discussed above.

### A. Router Architecture

The block diagram of the router architecture implementing the FUL+LB strategy in a mesh-based network topology is shown in Fig. 8. The top part of the picture shows the top level view of the router, whereas the bottom part shows block diagrams of the modules which implement routing function and selection function associated with each input port. If the current input link is partially faulty, the parts of the incoming flit are reassembled by the TFM RX module. In the other

\(^1\)For the sake of clarity, we have used a different symbol for representing selection probabilities computed with LB strategy. $Pr^{(LB)}(l, i, l_i, l_o)$ refers to a given node, a given input link, and a given destination.
case, the TFM_RX module is bypassed. The routing module, based on the destination in the header flit, returns the set of admissible output links where the packet can be forwarded. The routing module also returns the selection parameters associated with that destination. This information is used by the selection module to select one of the admissible output links. The selection policy is based on the current traffic condition which is estimated by the traffic condition estimator block. It uses the information regarding the number of free slots available in the FIFO buffers of the neighboring routers (signals \( *_{fs} \)) in accordance with discussion in Section VI-B. Finally, after the arbitration and crossbar operations, the flit is forwarded toward the selected output link. If the link is partially faulty, the TFM_TX module is used to split it in sub-flits.

The routing function is implemented by means of a routing table. The routing table is addressed by the destination address. An entry of the routing table contains the following fields: \( L_{ao} \), \( ERM \), \( Pr_{LB} \), and \( Pr_{FLB} \). \( L_{ao} \) encodes the set of admissible output links that can be used to reach the current destination. If we consider the west input port, \( L_{ao} \) is a four bit field whose bits indicate which of the output ports between north (N), east (E), south (S), and local (L) can be used to reach the current destination. \( ERM \), \( Pr_{LB} \), and \( Pr_{FLB} \) are the parameters used by the selection function. The number of bits used for encoding them will be discussed later in this section.

The selection block uses this information to select one of the possible directions in \( L_{ao} \). It uses different selection policies based on the current traffic condition defined by the \( TFM_{/high} \) signal.

The block diagrams of the modules for flit splitting (TFM_TX) and reassembling (TFM_RX) are shown in Fig. 9(a) and (b), respectively. A 64-bit link is considered to be partitioned into four regions of 16 bits each. A 4-bit register (REG4_FM) is used to mark which regions can be used for a flit transfer. For instance, if REG4_FM contains 0101, it means that to send a flit, only link lines belonging to region 2 (bits 16 to 31) and region 4 (bits 48 to 63) can be used. The fault degree in this example is 50%. In this case, the flit is transferred in two cycles: first the bits from 0 to 31, then that from 32 to 63. The TFM_RX module in the neighbor router reassembles the flits. The TFM_RX is aware of the sequence in which the flit parts are received thanks to the information stored in its REG4_FM register. Of course, the same status information is stored in the REG4_FM registers in both the TFM_TX and TFM_RX modules of two neighboring routers.

We assume that each manufactured NoC chip is tested off-line and its faults are characterized [24]. We propose that all the identified faults of the chip are stored in an on-chip one time programmable ROM (OTPROM). We assume that various TFM_TX and TFM_RX register pairs can be loaded from this OTPROM at the time of booting the NoC chip. The detailed mechanism for achieving this is beyond the scope of this paper. We are currently working on details of this solution.

B. Router Design and Analysis

The six different routers implementing the strategies discussed in Section V have been designed in VHDL and synthesized using Synopsys Design Compiler and mapped on a 90 nm technology library from TSMC. The FE router uses a five-stages pipeline with the following stages: FIFO, routing, selection, arbitration, and crossbar. In the other routers, an additional pipeline stage (TFM) is introduced to perform flit splitting/reassembling operations. However, a forwarding mechanism can be used to exclude this stage in the case the outgoing link is fault free. FIFO buffers depth is four flits, with 64-bit flit size.

Two bits have been used to encode \( Pr_{LB} \); whereas three bits have been used for \( Pr_{FLB} \). With two bits it...
is possible to classify fault degree into four levels which is
inline with the number of partitions used by the TFM modules.
With regard to $P_{LB}$ and $P_{FLB}$, we found that using more
than three bits does not translate into appreciable improvement
of performance. This can be observed from Fig. 10 which
shows the variation in average delay for butterfly traffic when
$P_{FLB}$ is encoded using 2, 3, 4, and unlimited (double precision
floating point) number of bits.

Fig. 11 compares different routers in terms of area, delay,
and power.

With regard to the power analysis we proceeded as follows.
We developed a testbench to stimulate the input ports of the
routers with packets of different sizes (from two to 16 flits) and
random destinations. We simulated the netlist collecting the
toggle information for every net in the router. Then, we used
the information gathered from simulation (switching activity)
to annotate each net in the design. Finally, we used Synopsys
Design Compiler to get power figures.

In terms of area, the overhead spans from 15% to 21%. It
is mainly due to the increase in width of routing tables as
they need to store the selection probabilities required by the
different selection functions. The TFM_TX and TFM_RX
modules contribute to the router area by only 6% and 5%
on average, respectively. However, it should be pointed out
that such overhead is specific for the considered configuration.
That is, if FIFO depth, or flit size increase, the area overhead
decreases. For instance, in this analysis we consider four-flits
FIFOs but if we increase FIFOs depth to eight flits, the area
overhead due to TFM_TX and TFM_RX reduces to 4% and
3%, respectively.

Fig. 12(a) shows the area breakdown. As can be observed,
area is dominated by FIFO buffers and control logic.

We targeted the design to work at a clock frequency
of 1 GHz. The maximum clock frequency supported by the
router is not affected by the introduction of the new selection
strategies and the TFM logic. The maximum speed is limited
by the access to the routing table and by the control logic.

With regard to power dissipation, an increase of 5% to 8%
is observed. Fig. 12(b) shows the power breakdown. As can
be observed, power dissipation is dominated by FIFO buffers
and control logic. It should be pointed out, however, that such
power values are representative of the worst case in which all
the outgoing/ingoing links of the router are partially faulty and
all TFM modules are enabled. In a large network with only a
small percentage of links being faulty, the power increase will
be negligible.

IX. EVALUATION AND COMPARISON OF VARIOUS ALGORITHMS

A. Simulation Platform and Setup

In this section, we compare different strategies described
in Section V. The evaluations were made on a 8 × 8 mesh
network in which faults were randomly injected as shown
in Fig. 13. The network is affected by 25 full link faults
(about 10% of the links), and 15 partial faults (about 7% of
the links). For the latter, fault degree is randomly generated.
Number of faults due to manufacturing defects are likely to be
lower than assumed in our setup. This is done to highlight the improvement possible while using strategies which try to use partially faulty links and try to balance load in various links. The setup especially demonstrates the importance of advance strategies for handling heterogeneous topologies and network with different link capacities.

Wormhole switching was used with a packet size randomly distributed between two and 16 flits, and routers with input buffers size of 8 flits. The values thresholds $T_1$ and $T_2$, discussed in Section VI-B, have been set to 4 and 2, respectively. We have done numerous simulation experiments evaluating several $T_1$ and $T_2$ combinations under different traffic scenarios for different buffer sizes. Based on these experiments, we observed that for the best results the threshold values correlate well with the FIFO size ($fs$). Precisely, we observed that, on average, the best results are obtained when $T_1 \approx fs/2$ and $T_2 \approx fs/4$. We use the source packet injection rate ($pir$) as load parameter with Poisson packet injection distribution. For each load value, latency values are averaged over 100,000 clock cycles after a warm-up session of 10,000 clock cycles. The 95% confidence intervals are mostly within 2% of the means. Simulations were carried out with NoC simulation platform Noxim [25].

B. Performance Analysis

Fig. 14 shows the delay variation and energy variation under uniform random traffic. As can be observed, the exploitation of partially faulty links operated by FU, FUL, FE+LB, FU+LB, and FUL+LB strategies increases the saturation $pir^2$ by up to 16% with respect to FE, and reduces average delay by 73%. In terms of energy, a reduction of 15% is observed between FE and FU/FUL, and additional 4% of saving can be gained when the load balancing technique is used.

Due to space limitation, we do not report the detailed results for the other traffic scenarios. However, taking FE as the baseline implementation, a summary of the improvements in terms of percentage increase in saturation $pir$, reduction of both average delay and energy consumption, is shown in Fig. 15. With regard to saturation $pir$, an increase of up to 38% is obtained with FUL+LB under bit-reversal traffic. Averaging on all types of traffic patterns, increase in saturation $pir$ ranges from 12% to 22% from FU to FUL+LB. However, for butterfly traffic, increase in saturation $pir$ is quite low (less than 5% on average). This is due to the fact that the performance increase for the strategies using partially faulty links as compared to those not using them (like FE and FE+LB) is significant only in the cases when such additional resources are available.

Nevertheless, butterfly traffic is characterized by few and short communications, mainly using vertical channels. This, along with the fact that the particular topology used in the experimental setup contains a few vertical partially faulty links, justifies the results.

Reduction in average delay is much evident. More than 85% reduction in average delay is observed when FU+LB and FUL+LB are used under bit-reversal traffic. Averaging on all traffic patterns, percent average delay reduction ranges from 41% to 55% from FU to FUL+LB.

In terms of energy, even if routers implementing the new selection strategies are more power hungry than FE router, a reduction in energy consumption is however obtained. This
...is due to the fact that although new routers dissipate more power, they work for less time to handle the same amount of traffic. However, in some cases, the increase of router power dissipation is not enough counterbalanced by performance improvement. It is the case for butterfly traffic in which energy consumption increases. However, averaging on all the traffic the proposed strategies result in an improvement in energy consumption with a saving ranging from 6% to 12%.

C. Analysis for Varying Faults Distribution

The results presented in the previous subsections have been obtained considering a specific faults distribution as shown in Fig. 13. For the sake of generality, in this subsection we analyze the performance variation for varying fault distribution. Precisely, we consider two faults distribution named random and clustered. In random distribution, faults are uniformly injected over the network links. In clustered distribution, the first fault is randomly injected and the subsequent faults are injected \( h \) hops far away from the first one with a probability which is inversely proportional to \( h \). As soon as a fault is injected, its fault degree is randomly determined.

Fig. 16 shows the variation of average delay for FE and FUL+LB under uniform traffic scenario for different percentage and distribution of link faults. As expected, the average delay increases as percentage of link faults increases. However, the increase is much slower for FUL+LB. For randomly distributed link faults, average delays increase less than 25% for FUL+LB whereas it increases more than 540% for FE when links fault percentage increases from 0.5% to 5%. When link faults are clustered, the difference between the two strategies becomes even more pronounced. FE is not able to sustain more than 2% link faults whereas FUL+LB continues without any significant increase in delay (less than 30%) up to 5% of link faults.

D. Case Study: MultiMedia System

In this subsection, we assess the proposed approaches for a real case study. We consider a generic multimedia system (MMS) which includes an H.263 video encoder, an H.263 video decoder, an MP3 audio encoder, and an MP3 audio decoder [26]. The communication graph of MMS is depicted in Fig. 17(a). It has been partitioned into 40 distinct tasks which have been assigned and scheduled onto 25 selected IPs. The IPs have been mapped using a modified version of the multiobjective mapping approach presented in [27] in which we considered communication cost [26] and paths diversity [28] as optimization metrics. From the obtained Pareto set, we selected the tradeoff solution in the middle of the Pareto front. Such a mapping is shown in Fig. 17(b). Please note that for this mapping a minimal fully adaptive routing is deadlock free. We used paths diversity as additional optimization index as having multiple disjoint paths for any communicating pair is essential in the context of fault tolerance.

Fig. 18 shows the ratio between the actual bandwidth and the demanded bandwidth for the top ten communications which demand more bandwidth under different percentage of faulty links. We assumed network links providing a bandwidth of 300 Mb/s and injected faults with a fault degree of 0.5. As can be observed in the graph associated with FE, there are no bars for some communications at some percentage of link faults. This happens when an injected fault affects a link used by a minimal path which is unique for a given communication. In this case that communication cannot be realized anymore. On the other side, using the proposed strategies (FU, FUL, and +LB variants) such communication can still be realized although some of the links of the path provide less bandwidth than the nominal link bandwidth. Results in Fig. 18 show that link faults lead to reduction in available bandwidth for certain communicating pairs in the system. These reductions are different for different pairs and different strategies. The reductions are minimum for...
Fig. 18. Ratio between the actual bandwidth and the demanded bandwidth for the top ten communications which demand more bandwidth under different percentages of faulty links.

Fig. 19. Percentage video resolution reduction under different percentages of faulty links.

the FUL+LB strategy. There are various scenarios for the consequences of this reduction in performance.

1) In the first scenario, the application is run on the platform with a slightly reduced performance. For the considered case study, a NoC platform with 5% faulty links can be used if a 30% reduction in video resolution is acceptable. Fig. 19 shows the resulting percentage video resolution reduction corresponding to different percentages of link faults.

2) In the second scenario, if reduction in performance is not acceptable, then one can try a different mapping of application on the platform, which is aware of the presence of faulty links. Since NoC platforms are generally "over-designed" for both communication as well as computational performance, there is a good chance that the re-mapping can achieve the required performance.

3) The third scenario when reduction in performance is not acceptable, and re-mapping is not an option or does not result in the required performance, there is no choice except to reject the chip for the particular application. The chip could be used for some other application with lower requirements.

Another key difference between FE and the proposed strategies is that using FE when a fault affects a link, the traffic volume allocated on the routing paths containing the faulty link must be reallocated on other routing paths. So, the communication load on such paths increases such that the aggregated bandwidth on some links exceeds the link capacity. From Fig. 18(a), it can be observed that there are many dead communications (i.e., communications that cannot occur for the lack of routing paths between the source and the destination node). This is due to the fact that FE prohibits the use of partially faulty links. So, two cores mapped onto two neighbor nodes connected by a partially faulty link cannot communicate as we are assuming minimal routing. This situation will occur often as the mapping strategy used to map the cores into the NoC [27] tries to put communicating cores close to each other. Thus, with a high probability, cores which communicate frequently are mapped as neighbor nodes (one hop distance). The tradeoff between mapping frequently communicating cores close to each other for minimizing communication latency or mapping them in such a way so as to maximize the routing paths diversity (i.e., maximize the communication robustness) has been analyzed in [29].

X. CONCLUSION

In this paper, we have made a case for not discarding NoC chips which have minor manufacturing faults. Such chips, with massive computational power and interconnection infrastructure but with a few faults, may suffice for design of many products. In this paper, we have proposed modification to the basic router architecture to tolerate link faults and still provide routing functionality but with slightly degraded performance. We have proposed a series of efficient routing strategies to use NoC systems with partially faulty links. These strategies combine the knowledge of communication requirements of applications and capacity of various communication links to give high routing performance by avoiding traffic congestion in the network. We evaluate and compare our proposed strategies for average latency and energy consumption for various communication traffic patterns. There is sufficient evidence that the use of partially faulty link can significantly enhance network performance for higher loads compared to not using the partially faulty links at all. Results show that our most intelligent strategy (FUL+LB) can reduce the average delay by almost 50% for high traffic loads just below saturation point. Almost 20% reduction in energy is also achieved by the most intelligent strategy. The routing strategies proposed are topology agnostic and are also very suitable for design of networks in which link capacities can
be controlled programmed to optimize power consumption in routers or links by controlling router clock frequency.

We will like to record that the presented area and power figures in the paper are derived using a pre-placement and pre-route analysis, thus, the derived values of these parameters are not very accurate and can only be used for relative comparison of different strategies. We feel this comparison is valid since the base architecture of the router is common in all strategies. The more advanced strategy are implemented by different amount of augmented logic and, for this reason, we assumed that an increase in area and power consumption is proportional to amount of augmented logic.

REFERENCES


[8] S. Borkar, F. Angiolini, D. Bertozzi, and L. Benini, “Fault tolerance that an increase in area and power consumption is proportional to amount of augmented logic.

Maurizio Baleri (M’06) received the M.S. and Ph.D. degrees in computer engineering from the Indian Institute of Technology Delhi, Delhi, India, in 1994 and 1999, respectively.

Since December 2003, he has held a research con-tract as an Assistant Professor with the Department of Information and Telecommunications, University of Catania, Catania, Italy, in 1999 and 2003, respectively.

Dr. Baleri serves on the Editorial Board of very large scale integration (VLSI) design as an Associate Editor since May 2007. He has served as a Guest Editor for the VLSI Design Journal - Special Issue on Networks-on-Chip in 2008, and as a Guest Editor for the International Journal of High Performance Systems Architecture - Special Issue on Power-Efficient, High Performance General Purpose and Application-Specific Computing Architectures in 2009. He serves as the Technical Program Committee Chair for the following IEEE/ACM International Conferences: IETAS, CODES-ISSS, ESTIMedia, SOCC, VLSCI, and SITISE. He was also the Co-organizer of international workshops on network-on-chip architectures in 2008 and 2009.

Shashi Kumar (M’90-SM’08) received the B. Tech degree in electrical engineering, the M.Tech. degree in computer science and engineering, all from the Indian Institute of Technology Delhi, Delhi, India, in 1974, 1976, and 1985, respectively.

He joined the Department of Computer Science and Engineering in 1986 as an Assistant Professor, became an Associate Professor in 1991, and a Full Professor in 1997. Since December 1999, he has been a Professor of Embedded Systems with the

Department of Electronics and Computer Engineering, School of Engineering, Jönköping University, Sweden. He was a member of the team which was the first to propose the idea of packet switched networks for on-chip communication and coined the term network-on-chip (NoC), in 2000. He has interest in various aspects of NoC design, including NoC topologies, quality of service issues in NoC communication, NoC architectural modeling and evaluation, application-specific NoC architecture design, mapping applications to NoC platforms, and testing of NoC. His research interests include system-level modeling and synthesis, parallel architectures and algorithms, reconfigurable computing, and heuristic search algorithms.

Prof. Kumar served as the Guest Editor for the special issue on the Network-on-Chip for VLSI Design Journal, in 2008. He was also the Co-organizer of international workshops on network-on-chip architectures in 2008 and 2009. He has served in the Technical Program Committees of various international conferences, including Design Automation and Test in Europe Euro-Micro Digital System Design, and the NoC Symposium.

Vincenzo Catania received the M.S. degree with Honors in electrical engineering from the Università di Catania, Catania, Italy, in 1982. Up until 1984, he was responsible for testing microprocessor system at STMicroelectronics, Catania, Italy. Since 1985, he has cooperated in research on advanced computer architectures and computer networks with the Dipartimento di Ingegneria Informatica e delle Telecomunicazioni, Facoltà di Ingegneria, Università di Catania, where he is currently a Full Professor of Computer Science. Since November 2006, he has served as the Director of the Department of Computer Science and Telecommunications Engineering, Università di Catania. He is the author of more than 200 articles on international journals and conference proceedings and holds two patents. Currently, his research focuses on pervasive embedded systems, network-on-chip architectures, and mobile terminal platform and services.