# Enhancing the Security of Time-Division-Multiplexing Networks-on-Chip through the Use of Multipath Routing

Radu Stefan
Eindhoven University of Technology
R.Stefan@tue.nl

Kees Goossens
Eindhoven University of Technology
K.G.W.Goossens@tue.nl

## ABSTRACT

After gaining popularity as a method of authentication in the form of smart cards, electronic security mechanisms are making their way into the domain of embedded domain with the goal of protecting Intellectual Property or for Digital Rights Management. A key role in implementing security at chip-level is played by the interconnect, which has the task of providing and regulating the flow of data between an increasing number of on-chip elements, not all of which can be considered trustworthy. Networks-on-Chip are emerging as a scalable solution for modern on-chip communication. In this study we aim to improve NoC security by forcing the messages to be routed on multiple disjoint paths, optionally in a non-deterministic manner. We implement our proposal and find it to have a reasonably low cost in terms of hardware area, although potentially having a larger overhead in terms of allocated bandwidth.

## 1. INTRODUCTION

The level of integration achieved by the current silicon technology presents the designers with significant challenges regarding the on-chip communication infrastructure required to connect the increasing number of on-chip IPs. At the same time, the security of electronic circuits and applications increasingly becomes a concern. Networks-on-Chip [6, 3] have emerged as a promising alternative to the traditional interconnects, offering scalability, reusability and easy communication across clock domains, but have yet to prove their worthiness from the security point of view.

Security is desired in a variety of applications. Users require that their personal data and communications are kept secure, chip and software manufacturers want to prevent unauthorized replication of their products and content owners aim to prevent illegal distribution of their content. Security applications find promising solutions in the form of secure chips.

Cryptographic keys are stored securely on chip and are never communicated outside in a readable form. The inner workings of the chip are scrambled in order to prevent reverse-engineering and tamper detection elements are employed to disable the device if intrusive techniques are used to probe the chip. Nevertheless, this solution is not without pitfalls. Attackers find innovative ways to disable security measures, to read protected data or to reverse engineer manufactured IP.

The on-chip interconnect is a particularly sensitive element from the security point of view. Unlike the external signals it is usually too expensive to encrypt. The long wires placed in the upper metal layers, having powerful drivers are particularly susceptible to probing [13] and side-channel attacks [19, 16]. [20] shows how top-metal lines are easily observed using voltage contrast attacks.

The software running on embedded systems represents another potential vulnerability. A malicious application may be disguised as a game for mobile phones, or even as non-executable, corrupted media that takes control of the application whose task was to display it. Font, video and graphics renderers have been recently targeted by such attacks.

In the context of the Networks-on-Chip this type of attack becomes particularly dangerous once a malicious application gains control of one of the network nodes. Although Networks-on-Chip are more secure than busses, where each transaction is broadcasted to all connected elements, it may still represent the target of combined software and fault-injection [1, 2] attacks, where information is either prevented from travelling through the network or rerouted to the wrong location.

Another attack can be envisioned as a combination of software and power analysis attacks. Because the network is interleaving the different communication streams, when the attacker has control of one of these streams, he could choose what symbols to transmit and he could measure the amount of switching that takes place between the chosen symbols and a secure stream.

We attempt to thwart such attempts by introducing more non-determinism in the communication system in the form of multi-path routing. In our proposal, communication channels designated as secure are routed over multiple disjoint paths. The path selection can be either according to a predefined schedule, or it can be performed randomly at runtime. We implement our proposal in the context of Æthereal and analyze the additional hardware cost and the cost in terms of allocated bandwidth.

## 2. RELATED WORK

The security of the on-chip communication interconnect has already been addressed in several studies, and solutions for improvement have been proposed.

In [12], the authors provide a comprehensive overview

*NoCArc '11,* December 4, 2011, Porto Alegre, Brazil

of the security risks in Embedded Systems Design. Software, physical, side-channel and fault-injection attacks are detailed and solutions that were proposed in the literature against these attacks are presented.

A Security Enforcement Module implemented at the level of the AMBA bus is proposed by [5]. It achieves protection against malicious embedded software or external hardware connected to the system by filtering memory access requests based on their address and possibly data.

In [9] the authors propose a modular monitoring system integrated with the NoC at the level of the network interfaces. The system consists of Probes which are used to detect misbehavior of processing elements that are trying to access data, an infrastructure for sending security events to a centralized Network Security Manager which can then disable the offending element. The solution targets both illegal access and denial-of-service attacks.

An energy efficient bus scrambling system based on modulo 2 addition with a pseudo-random variable is presented in [4]. The authors attempt to improve power consumption by reducing the otherwise large number of transitions.

A method for protecting hardware IP by using bus-scrambling with bit-permutations and substitutions is proposed in [17]. Circuits using this feature will need to be unlocked post-production through an encrypted handshake.

A source routing mechanism that makes spoofing of the source node impossible is proposed in [7]. The technique also ensures that it is not possible to insert packets causing livelock. The authors detail a configuration infrastructure for securely bootstrapping the device.

In [10] the authors propose the use of Security Wrappers providing encrypted transmission for the distribution of sensitive information like cryptographic keys. The system employs a centralized manager called a Key Keeper.

Our proposed technique is orthogonal to the previously presented solutions and can be used in combination with them for an additional increase in security.

For other applications different than security, multipath routing in NoCs has been previously proposed in [15, 18].

# 3. HARDWARE ARCHITECTURE

We implement our proposal in the framework of the Æthereal Network-on-Chip. Æthereal is a connection-based TDM (Time Division Multiplexing) circuit-switching network, whose characteristics make it highly desirable both from the security point of view and for ease of implementation of the multipath routing strategy. In particular it provides contention-free routing [11] preventing communication streams from interfering with one another. In the following, we will explain what benefits are obtained through the use of the Æthereal network and what enhancements we propose.

## 3.1 Intrinsic security features of the Æthereal implementation

Æthereal offers connection-based services which behave transparently like FIFO queues between a source and a destination, with known delay and bandwidth. In order to achieve this, the bandwidth of each link is split in the time domain into allocation units called timeslots, as in a typical scheme of Time-Division-Multiplexing. Each communication stream receives exclusive use of a number of timeslots on each link on its path between source and destination.

The packets never arbitrate inside the network and never wait for each other, but instead they are immediately forwarded into the slot that has been reserved for them. This scheme provides guarantees regarding the delay and bandwidth of each communication stream, non-interference of streams and freedom from deadlock at the network level. These qualities can be useful in avoiding timing attacks or denial-of-service attacks.

When considering the security of an embedded system or application, one has to take into account all the potential interaction of elements inside the system and the entire sequence of planned and unplanned events starting with system power-on, changes of state like going into and coming out of stand-by and proper system shutting down.

The configuration of the Æthereal network and surrounding busses is controlled by configuration ports. The configuration ports allow in the case of busses setting the memory address ranges a slave will respond to, and in the case of network interfaces the tables of paths and timeslots that will be used for communication (Figure 1). Initially, a single IP, commonly called the “Host” has access to the configuration port of its own Network Interface. The address ranges of its own local bus are also hard-coded to the address of the local and remote configuration ports. The host can then open up connections to the configuration ports of other network interfaces within the system (1,2) and set up connections between different IPs (3).

## 3.2 Proposed Implementation

We enhance the existing Æthereal implementation [11], by adding the capability for multi-path routing. In its most common form, Æthereal employs source routing of packets, which means that the route of each packet is completely described by the sending Network Interface. The route, stored in the packet header, is read from a table of routes inside the NI and contains the list of turns the packet must take at each hop in order to reach its proper destination. To ensure freedom of collisions, each connection is allowed to transmit only at specific moments in time given by a schedule also stored into the Network Interface.

Our modifications consist of an extension to the table of paths to accommodate the new extra paths for connections and a selection mechanism for those paths, figure 2

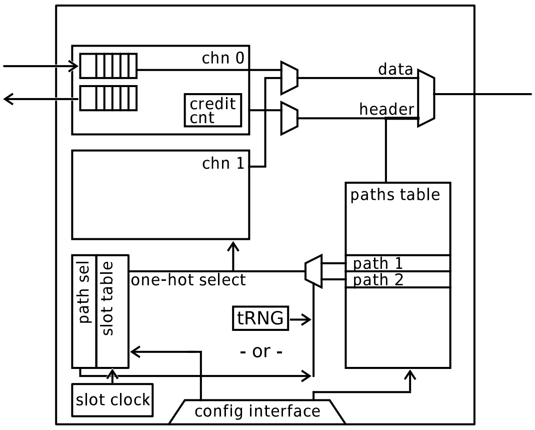

**Figure 2: Architecture of a network interface with support for multipath**

We envision two possible approaches for improving NoC security by using multipath routing. In the first approach,

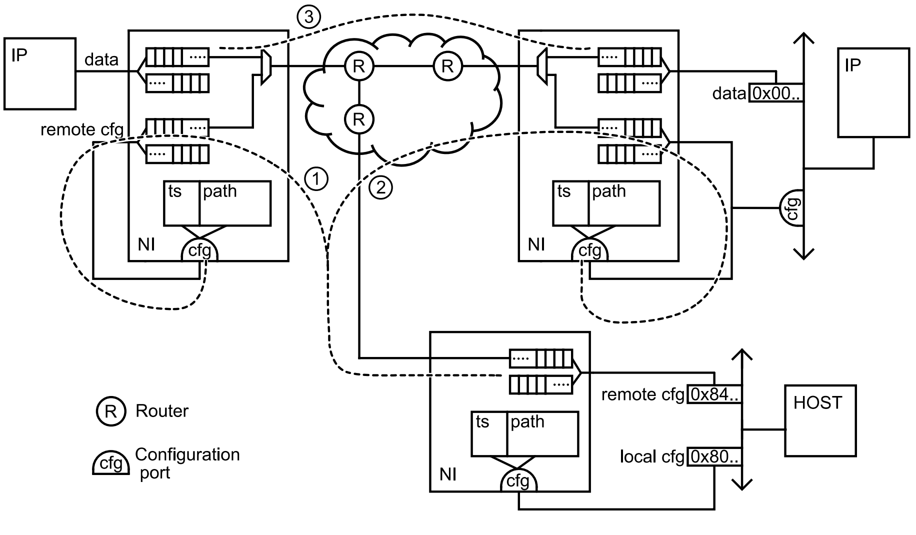

**Figure 1: Security in the Æthereal implementation**

the schedule is static and selecting one of the alternative paths is done entirely based on the position in the slot table at the moment of sending. The second option consists of dynamically selecting one of the paths at run-time based for example on a hardware random number generator. This non-deterministic behavior is valuable from the security point of view, but it has a disadvantage in that more slots need to be reserved for the same connection.

The two approaches are similar in implementation, the distinction consisting in the source of the path selection signal 2. For the static approach, the path is selected according to the slot schedule, while for the dynamic approach it is generated using a true or a pseudo random number generator. Pseudo-random number generators can be complex enough to make the sequence of paths unpredictable, but they may be susceptible to replay-type attacks.

We have verified our multipath implementation in FPGA, both using the static and “dynamic” routing. For the dynamic case, a pseudo-random number generator was used rather than a True Random Number Generator (tRNG). The test setup consists of a MicroBlaze processor, communicating over a 2x2 mesh network with a remote memory. The Network-on-Chip is configured so that both paths between opposite corners are used for the data transfer. Optical signaling through on-board LEDs is used to confirm the successful transaction. We have also verified in simulation that both paths are indeed used.

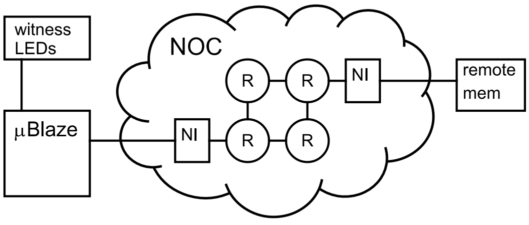

**Figure 3: FPGA verification setup**

# 4. SLOT ALLOCATION ALGORITHMS

In a previous study [18] it was shown that routing over several paths can provide a gain in terms of bandwidth, however, when using multipath for NoC security, the problem becomes more difficult for two reasons:

1. it is now required rather than optional to route certain channels over multiple paths
2. the paths need to be disjoint, otherwise interception at a single point would be possible

Although interception at a single point would still be possible inside the IP that produces or receives it or before the first router where the two paths eventually split in practice one NI and Router can be integrated in the local tile and the wires connecting them are much shorter than the long, inter-tile links.

## 4.1 Pathfinding algorithm

In the Æthereal framework, the routing of communication channels and the selection of TDM timeslots is typically performed at design time. Although it would be possible to perform this computation at run-time, this is not within the scope of this study.

As previously mentioned, in our target implementation, the bandwidth of each link is split into a number of TDM timeslots, which are statically allocated at design time to communication channels. Not all communication channels need to be active at the same time, and a slot can be used by multiple channels if the channels are mutually exclusive. Channels are grouped into usecases and the choice of which usecases are allowed to run in parallel is left to the system designer [14]. At this point however, we will not be delve into the details of sharing timeslots and will assume that, once reserved, a timeslot cannot be used by another channel.

Because global optimization algorithms would be too computationally intensive, channels are allocated one at a time, marking the slots as reserved after each allocation.

We will analyze the allocation process for a single communication channels. The allocation sees on each link marked slots which have been reserved by previous allocation, and must find a path to the destination which avoids the marked slots. Because in the Æthereal implementation [11] flits are not allowed to stall at the intermediate routers, they are forwarded over the next link in the immediately following slot (Figure 4).

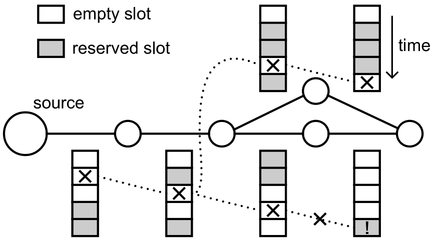

**Figure 4: Slot allocation, only possible when slots are not already reserved**

The pathfinding algorithm has thus a difficult task, the one of selecting from the multitude of paths the one that provides the best slot alignment. For both the single-path and multi-path algorithms, we use a depth-limited exhaustive search with a few enhancements.

In the first phase, the distance from every router to the destination node is computed with a simple one-to-all pathfinding algorithm applied in reverse. We use for this step the Dijkstra algorithm [8]. The algorithm takes into account that on some edges the bandwidth has already been depleted.

The exhaustive search algorithm starts with a bound on the total length of the path equal with the minimum path length to the destination computed by the Dijkstra algorithm. At each step, the length of the path already generated plus the remaining distance to the destination is not allowed to exceed the bound, in other words, each step must take us closer rather than farther from the destination. If a solution is not found, the bound is incrementally increased, thus allowing us to take first one, then more steps in the "wrong" direction.

To illustrate, consider Figure 5. In this figure, an allocation is attempted between the local NIs of routers R20 and R03. For demonstration purposes, let us assume that the links R01-R02 and R21-R22 do not have any bandwidth left and have been discarded from the computation. The distances from the destination node computed by the Dijkstra algorithm are written in the bottom left corner of each router.

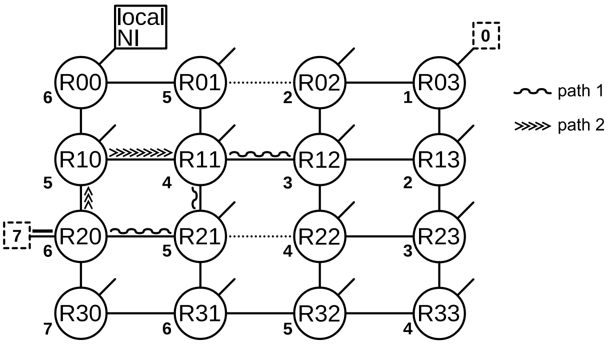

**Figure 5: Exhaustive path search, enhanced with distance-to-destination traversal order.**

When constructing Path 1 and visiting R20, the backtracking algorithm needs to choose the next step from the neighbors of R20. Only two of its neighbors are labeled with decreasing distances, R10 and R21. The algorithm will pick one of them and will continue the search. The other will be examined later, when returning from the initially chosen branch.

For constructing the multi-path solution, a similar algorithm is employed, but now the two paths are constructed in parallel, adding one link to each, while avoiding using the same link for both. Between a Network Interface and a Router this cannot be avoided though, as a single link exists. If one path reaches the destination before the other does, the search is continued by adding links only to the remaining path. Constructing the paths in parallel rather than one after the other provides us with early feedback on unfeasible solutions. For example in Figure 5, when the second path arrives at node R11, it cannot continue on a decreasing path to destination as the only link to a node with shorter distance is already blocked by path 1.

Once the destination is reached, we attempt to allocate a subset of the available slots on each path so that the bandwidth constraints are met.

In the dynamic mode, each of the paths must be able to support the entire channel bandwidth, while in the scheduled mode the bandwidth is divided between the paths. Depending on what resources are available for allocation the division is made arbitrarily between the two paths with the restriction that the bandwidth on each path should be greater than 0.

When the length of the two paths is not the same, guard slots (Figure 6) are inserted in order to prevent packets sent over the short path from overtaking the packets sent on the longer path. The guard slots are not actually reserved, thus they do not consume any of the link bandwidth, but the short path of the current connection is prevented from using them. A number of guard slots equal to the difference in length between the paths is inserted after the slots belonging to the long link.

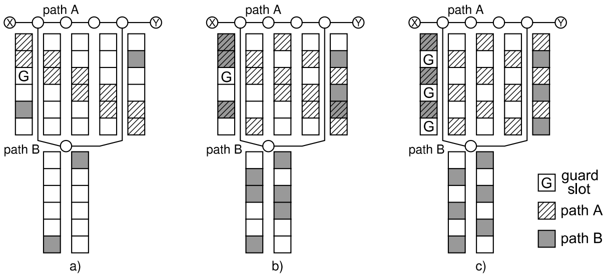

**Figure 6: Allocated slots, a) static, b) dynamic grouped, c) dynamic sparse.**

As it can be seen in figure 6, the dynamic (non-deterministic) mode requires the reservation of more network resources. When one connection uses consecutive slots, we have an additional choice of allowing or not allowing a path switch within the group of consecutive slots (we call this the grouped and the sparse mode). The grouped mode is more efficient, as the Æthereal network does not require header retransmissions for flits belonging to the same channel and traveling on the same path. Differences in path length also incur the largest overhead in the case of the sparse mode.

We have tested our algorithms on network configurations using the mesh topology, with sizes ranging between 4x4 and

6x6 and found them to produce suitable results, as it will be presented in Section 5.

## 5. EXPERIMENTAL RESULTS

The overhead of the proposed method consists of both additional hardware requirements and an increased difficulty in allocating channel bandwidth spread on two disjoint paths.

In order to determine the additional hardware area requirement, we have modified a network interface from the Æthereal network implementation to support the multipath feature. In a real application not all data streams need to be treated equally from the security point of view. For example, communication flows carrying encryption keys would be routed on secure paths, but the same should not be required for high-bandwidth multimedia streams. As a consequence our model, implemented in VHDL, is parameterizable and it allows us to specify the number of connections that are going to be routed on multiple paths. The model, which has also been tested in FPGA, has been synthesized in TSMC 90nm technology using the Synopsis tools. We found that the increase in terms of hardware area is roughly linear to the number of channels that are desired to be secure, up to a maximum of 2.8% when all channels are routed using multiple paths. Figure 7 presents the cost increases for a Network Interface supporting 4 communication channels.

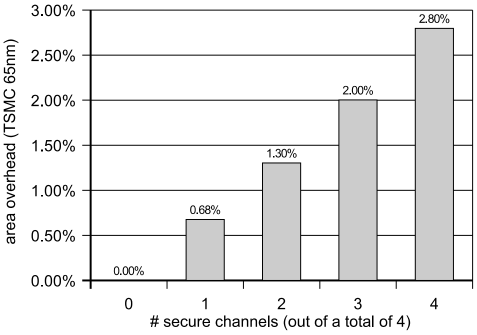

**Figure 7: Increase in Hardware Area**

In [18] it was shown that the multipath approach can actually improve bandwidth allocation by better utilizing the available resources. In that study, multi-path routing was suggested for the situations when it provided a better allocation than single-path, and falling back to single-path was allowed if that was not the case. In the current case, multi-path is considered mandatory even when it produces a result worse than single-path and furthermore, the paths are required to be disjoint.

We analyze the efficiency of allocating bandwidth under randomly generated usecases by randomly selecting some of the channels to be used as secure channels and measuring frequency needed to support the allocation of all connection in respect with the proportion of channels that were selected to be secure. In all our tests two disjoint paths are required for the secure channels.

For the deterministic path selection, the results are presented in Figures 8, 9. The results are averaged over 10 usecases for each considered topology and each considered proportion of secure channels.

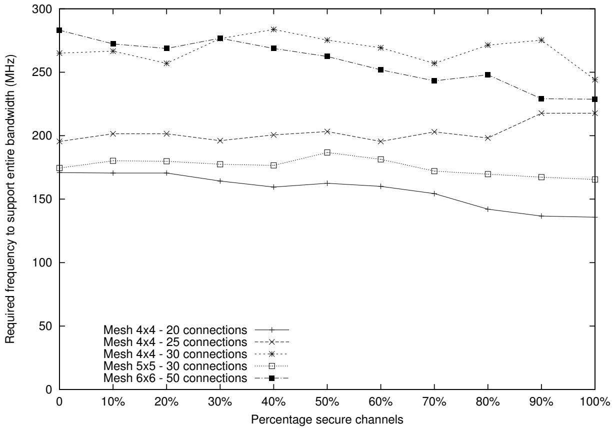

**Figure 8: Required frequency versus percentage of secure channels, deterministic path selection, secure channels allocated first**

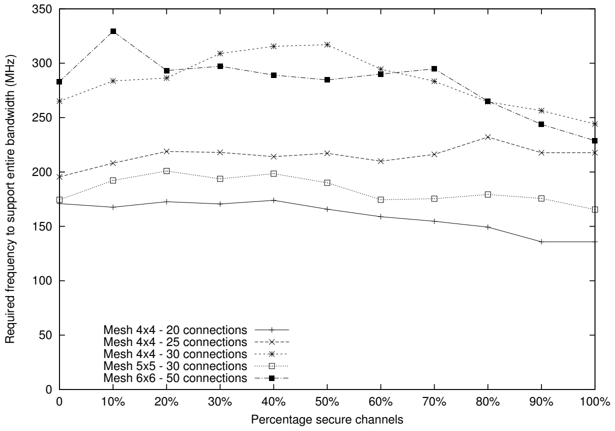

**Figure 9: Required frequency versus percentage of secure channels, deterministic path selection, secure channels allocated last**

As previously mentioned, the multi-path allocation can be either beneficial, reducing the required working frequency or detrimental, thus requiring a higher working frequency of the network in order to support the same total bandwidth. As a rule, we found that the combination of single-path and multi-path allocations together in the same usecase in similar proportions has the most detrimental effect, especially if the allocation of the secure channels is performed second. This is explained by the fact that single-path and multi-path have different requirements. Single-path performs best when the slot allocation is not fragmented. Multi-path can easily deal with fragmented slots but also produces more fragmentation. The secure multi-path requires bandwidth to be available on more than one path.

For the non-deterministic case, where the path can be selected dynamically at run-time during each flit cycle, the requirements are significantly higher. This is because the slots need to be reserved on multiple paths simultaneously. Because multi-path is more difficult in this case, it is always performed first.

We also have the option of forcing the stream to use the same path when consecutive slots are allocated to the same channel, thus increasing the useful payload, or a switch is allowed for each flit thus resulting in a sparse, less efficient allocation. The results are presented in Figures 10,11.

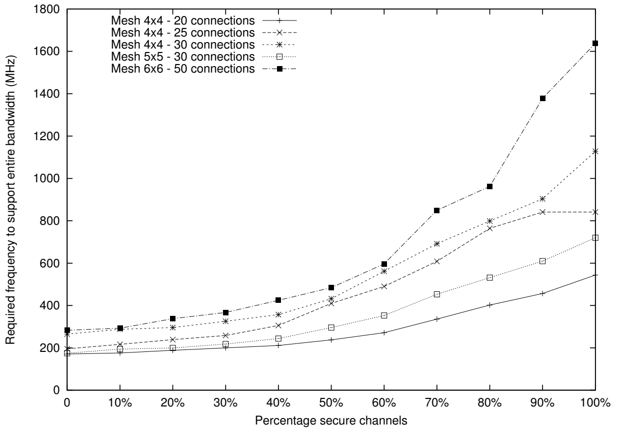

**Figure 10: Required frequency versus percentage of secure channels, path selected dynamically**

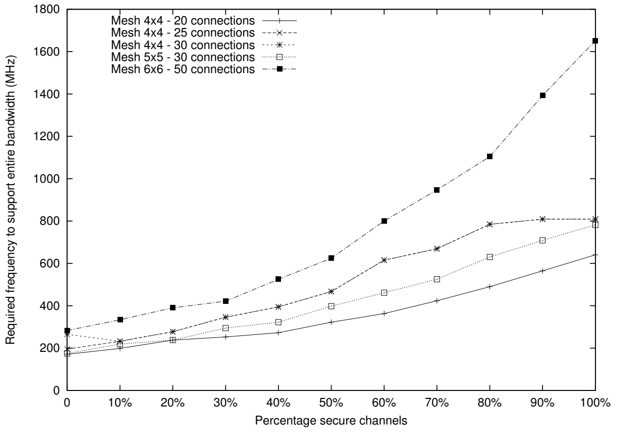

**Figure 11: Required frequency versus percentage of secure channels, path selected dynamically, sparse allocation**

Overall, we found that with a static path selection (periodically switching between the two paths) the allocation overhead is very low or in some cases there is even a gain in useful bandwidth/frequency, while in the dynamic mode the overhead can be as high as 584%. If the dynamic selection is still desired, it would be recommended to apply it for only some of the communication channels.

## 6. CONCLUSIONS AND FUTURE WORK

In this paper we present a method for enhancing on-chip security at the level of the interconnect through the use of multi-path routing. We implemented our design by adding multi-path functionality to the Network Interfaces of the Æthereal network on chip and we proposed routing algorithms performing an exhaustive search of disjoint paths in order to guarantee the required bandwidth at design time. Our multi-path approach is guaranteed not to generate out-of-order delivery of packets.

We provide a cost analysis in terms of hardware area, which is found to be in the range of 2.8% as well as frequency/bandwidth allocation across multiple configuration options. We also demonstrate our technology in FPGA.

In the future, we consider combining our proposed technique with other established techniques for improving on-chip security.

## 7. REFERENCES

[1] R. J. Anderson and M. G. Kuhn. Low cost attacks on tamper resistant devices. In *IWSP*, 1998.

[2] H. Bar-El, H. Choukri, D. Naccache, M. Tunstall, and C. Whelan. The sorcerer's apprentice guide to fault attacks. *Proceedings of the IEEE*, 94(2), 2006.

[3] L. Benini and G. De Micheli. Networks on chips: a new SoC paradigm. *Computer*, 35(1), Jan 2002.

[4] L. Benini, A. Galati, A. Macii, E. Macii, and M. Poncino. Energy-efficient data scrambling on memory-processor interfaces. In *ISLPED*, 2003.

[5] J. Coburn, S. Ravi, A. Raghunathan, and S. Chakradhar. SECA: security-enhanced communication architecture. In *CASES*, 2005.

[6] W. J. Dally and B. Towles. Route packets, not wires: On-chip inteconnection networks. In *DAC*, 2001.

[7] J.-P. Diguet, S. Evain, R. Vaslin, G. Gogniat, and E. Juin. NoC-centric security of reconfigurable SoC. In *NOCS*, 2007.

[8] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 1959.

[9] L. Fiorin, G. Palermo, and C. Silvano. A security monitoring service for NoCs. In *CODES+ISSS*, 2008.

[10] C. H. Gebotys and Y. Zhang. Security wrappers and power analysis for SoC technologies. In *CODES+ISSS*, 2003.

[11] K. Goossens, J. Dielissen, and A. Radulescu. Æthereal network on chip: Concepts, architectures, and implementations. *IEEE Design & Test of Computers*, 22(5), 2005.

[12] P. Kocher, R. Lee, G. McGraw, and A. Raghunathan. Security as a new dimension in embedded system design. In *DAC*, 2004. Moderator-Ravi, Srivaths.

[13] O. Kömmerling and M. G. Kuhn. Design principles for tamper-resistant smartcard processors. In *WOST*, 1999.

[14] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. De Micheli. A methodology for mapping multiple use-cases onto networks on chips. In *DATE*, 2006.

[15] S. Murali *et al.* A Method for Routing Packets Across Multiple Paths in NoCs with In-Order Delivery and Fault-Tolerance Guarantees. *J. VLSI-Design*, 2007.

[16] J.-J. Quisquater and D. Samyde. Electromagnetic analysis (EMA): Measures and counter-measures for smart cards. In *E-SMART*, 2001.

[17] J. A. Roy, F. Koushanfar, and I. L. Markov. Protecting bus-based hardware IP by secret sharing. In *DAC*, 2008.

[18] R. Stefan and K. Goossens. A TDM slot allocation flow based on multipath routing in NoCs. *MICPRO*, January 2010.

[19] K. Tiri. Side-channel attack pitfalls. In *DAC*, 2007.

[20] P. Tuyls and J. Walker. Conquering copycats: Attacks fail against hardware intrinsic security. White paper, August 2009.