Microprocessors and Microsystems 35 (2011) 130–138

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

A TDM slot allocation flow based on multipath routing in NoCs

R. Stefan [a,*], K. Goossens [b]

[a] Technical University of Delft, The Netherlands
[b] Technical University of Eindhoven, The Netherlands

ARTICLE INFO

Article history:
Available online 29 September 2010

Keywords:
Networks-on-chip
TDM
Multipath

ABSTRACT

Networks-on-chip have evolved as the natural solution for a scalable interconnect that can be automatically generated to suit the needs of the desired application. In this study we focus on improving the efficiency of on-chip networks using alternative routing strategies. We focus on a multi-path slot allocation method in networks with static resource reservations, in particular TDM NoCs. The simplicity of these networks makes it possible to implement this routing scheme without significant hardware overhead. Our proposed method, although displaying large variations between test cases, provides significant overall gains in terms of increased bandwidth or reduced working frequency or area. Our tests show that when using multipath routing the same communication requirements can be mapped on networks working on average at frequencies lower by 24.55% on average, while in individual cases the largest reduction was 60.04%. At the same time we are avoiding problems like deadlock and out-of-order delivery, commonly associated with multipath routing.

1. Introduction

Designing an efficient on-chip interconnect for systems-on-chip presents the engineer with multiple challenges. In addition to the raw performance requirements in terms of bandwidth and latency, the system designer has to consider possible interference between applications, the effect of crossing clock domains and guarantees regarding real-time system behavior. On the other hand the hardware requirements and power consumption of the interconnect have to be maintained at the minimum possible levels.

Networks-on-chip or NoCs represent the emerging paradigm for a scalable chip interconnect [1,2]. Among these, Circuit-switching NoCs [3–5] are an attractive solution as they are able to both isolate tasks from interfering with each other and they can provide communication channels with guaranteed bandwidth and latency. The time-division-multiplexing (TDM) technique employed in combination with circuit switching offers the means to divide the link bandwidth between channels with fine granularity and with discretionary budgets allocated to each channel.

We base our experiments on the Æthereal network [3,6]. Æthereal uses overall knowledge about the system behavior at design time to dimension the network and create allocations for each channel for each of the desired use cases. The allocation itself is performed by automatic tools which resemble in function the circuit routing tools, the main difference being that time is used as a degree of freedom in addition to space.

In the Æthereal implementation, the bandwidth of each link is split, in the time domain, into discrete allocation units called time slots. Typically, the entire bandwidth of each communication channel is allocated in one or more time slots along a single physical path. There may be the case though that no path is found that can satisfy the communication requirements. The designer is then forced to increase the hardware resources allocated to the network or its working frequency. As an alternative, it may be possible to divide the traffic of the channel that could not be handled using a single-path allocation over multiple physical paths.

In this study we evaluate the performance of an allocation flow which makes use of multi-path allocation in addition to the typical single-path algorithm. Communication channels are allocated one-by-one as in the traditional approach, but whenever the single-path search fails an allocation over multiple paths is attempted. This approach is shown to produce on average reductions in the working frequency of the NoC of 24.55% or alternatively can be used to reduce the size of the network necessary to support the communication requirements.

The rest of the paper is organized as follows. In the following section we present related work, and usage of multipath routing in other domains than networks-on-chip. Section 3 illustrates the hardware changes necessary to support our proposal. The TDM slot allocation algorithms are presented in Section 4. Experimental results are presented in Section 5 while the last section presents our conclusions.

* Corresponding author.
E-mail addresses: R.A.Stefan@tudelft.nl (R. Stefan), K.G.W.Goossens@tue.nl (K. Goossens).

0141-9331/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.micpro.2010.09.007

2. Related work

In large scale networks, multipath routing has already been in use for a long time, for example in Internet traffic engineering [7] The problem presents other challenges though than the small-scale networks found on-chip. Buffering is in general plentiful by comparison and in-order delivery is not a requirement, because the protocol stack implements reordering in another layer, before the data is delivered to the user.

The problem of multipath routing in networks with resource reservation ie. asynchronous transfer mode or ATM was studied by Cidon et al. [8,9], and was shown to provide a benefit in terms of connection establishing time, while having mixed results from the bandwidth point of view.

Multipath routing in NoCs has been previously proposed in [10], however, the method presented there requires a complex mechanism for ensuring in-order delivery. Sequence identifiers are assigned to the packets and an arbiter rearranges them in the proper order at the point where the different paths converge. In contrast, in our solution the entire schedule of packets in-flight is known at design time and in-order delivery can be ensured entirely by selecting proper insertion time for each packet.

Multipath routing is also found in the various forms of adaptive routing or other forms of non-deterministic routing [11], largely addressed by studies of multiprocessors and now also applied to networks-on-chip [12–16]. The target of these studies is mainly guaranteeing deadlock freedom while maximizing utilization, but without explicitly addressing the costs of in-order delivery.

Our study targets NoCs that support resource reservation using time-division-multiplexing, in particular the Æthereal network [3]. Our algorithm performs both routing and slot allocation. Routing and slot allocation in a similar TDM setup is also studied in [17,18].

The same technique can also be applied to other TDM networks described in the literature, like the Nostrum network [5], aSoC [19] and the TDM test delivery in [20], the more flexible TDM technique of [21] and perhaps also to networks using space division circuit switching like [22].

A solution for performing mapping, single path routing and slot allocation in the Æthereal networks is presented in [23,24] performs in addition topology selection but using another network model which does not require slot allocation. The authors of [25] propose a graph coloring algorithm to solve a slot allocation problem in a similar network implementation but with more relaxed timing constraints.

3. Hardware architecture

The Æthereal network-on-chip implementation that we use in this study employs a routing model called contention-free routing [3,26]. Under this model, arbitration is avoided at the router level by ensuring at design time that packets only travel through the network according to a fixed schedule, which does not allow conflicts. The usage of each link is divided into fixed size time slots and each connection receives exclusive use of a subset of these time slots.

For proper functioning of the TDM schedule it is necessary that network elements are synchronized with their neighbors at flit level and agree on what is the current flit position within the schedule. In a typical synchronous implementation, a flit is composed of three words and thus transmitted during three clock cycles. One cycle would be spent for crossbar traversal, one on link traversal and one in the input buffers of the routers. However, it is not necessary that the network is synchronous at clock cycle level, even between direct neighbors. While the routing decision and crossbar traversal has to be simultaneous for all inputs of the router, asynchronous input buffers can be used and the time spent in buffers can vary to compensate for clock skew and link traversal time, as illustrated in Fig. 1.

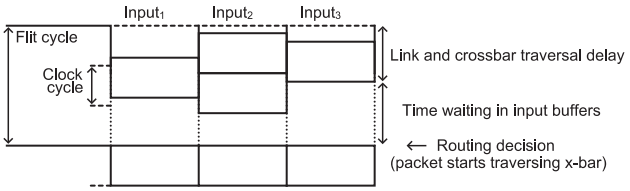

Fig. 1. Flit-level synchronization.

Two main approaches are possible for implementing routing in compliance with this schedule. One consists of storing the routing information in a distributed manner in all routers and network interfaces (NIs) and the other employs source routing for deciding the path each packet should take. In both cases the NI is responsible for injecting packets into the network only in the allowed time slots.

3.1. Distributed routing

In the distributed routing implementation [27] each router is synchronized to a global clock at flit level and contains a routing schedule which specifies to which of the outputs each input has to be routed during each time slot. Once the slot tables are configured into the routers, the destination of each packet is determined solely by its time of insertion into the network.

Under this implementation, no modifications to the hardware are necessary in order to support multipath routing. Programmed with the right schedules, the routers can lead the packets on a variety of paths to destination. There is no cost related to the number of different paths used by a single connection and there is an additional benefit in that no headers are necessary for routing the packets, thus increasing the size of the useful payload.

The absence of routing headers reduces the cost of the multipath approach, however in our tests we have conservatively assumed the presence of these headers.

3.2. Source routing

The entire communication is performed in a predetermined manner, usually computed at design time. At run-time the path taken by packets and the time when one connection is allowed to use the physical links are read from tables inside the network interface.

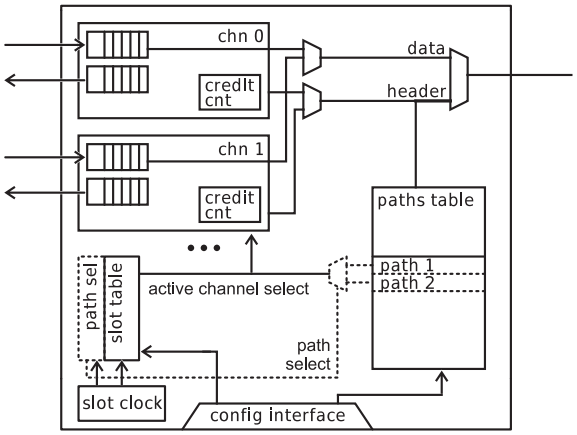

Fig. 2. Architecture of a network interface, the slot table dictates which channel is allowed to transmit at any given time; an additional table of paths selects the path to be used by the currently active channel.