Comparison of the Æthereal Network on Chip and Traditional Interconnects - Two Case Studies

Arno Moonen[1], Chris Bartels[1], Marco Bekooij[2], René van den Berg[2], Harpreet Bhullar[2], Kees Goossens[2,3], Patrick Groeneveld[1], Jos Huisken[4], and Jef van Meerbergen[1,5]

[1] Eindhoven University of Technology, The Netherlands
[2] NXP Semiconductors, The Netherlands
[3] Delft University of Technology, The Netherlands
[4] Silicon Hive, The Netherlands
[5] Philips Research, The Netherlands
a.j.m.moonen@tue.nl

Abstract. The growing complexity of multiprocessor systems on chip make the integration of Intellectual Property (IP) blocks into a working system a major challenge. Networks-on-Chip (NoCs) facilitate a modular design approach which addresses the hardware challenges in designing such a system. Guaranteed communication services, offered by the Æthereal NoC, address the software challenges by making the system more robust and easier to design.
This paper describes two existing bus-based reference designs and compares the original interconnects with an Æthereal NoC. We show through these two case study implementations that the area cost of the NoC, which is dominated by the number of network connections, is competitive with traditional interconnects. Furthermore, we show that the latency in the NoC-based design is still acceptable for our application.

1 Introduction

The integration of different types of cores like CPUs, DSPs, ASIPs and accelerators into a working system is a major challenge. The bottleneck in such multiprocessor architectures shifts from computation towards communication. Getting the right data at the right place at the right time will dominate the architecture. Currently busses and custom interconnects (point-to-point, crossbar switches) are often used, but with an increasing number of cores designed in technologies with decreasing dimension, they do not sufficiently address hardware problems (deep sub-micron VLSI design) and software problems (application programming). Networks-on-Chip (NoCs) tackle these problems and therefore are a better answer to the integration challenges.

First, *hardware problems*: NoCs help to answer some basic *deep sub-micron questions* because they structure the top level wires in a chip, and facilitate modular design [17]. Structured wiring results in predictable electrical parameters, such as crosstalk, etc. NoC interconnects are *segmented* and *multi-hop*. The advantage of segments is that only those segments are activated that are actually used in the communication. So only

those segments dissipate power. Multi-hop is needed because the transport delay from source to destination can become longer than the clock period.

Second, *software problems*: To reduce the programming effort proper transport level services have to be defined. In particular, networks on chip that offer *guaranteed communication services* (such as the Æthereal NoC [7] used in this paper) make systems on chip more robust, easier to design [8] and easier to program with a much lower non-recurring engineering cost. NoCs also provide concurrency, i.e. several transactions can be dealt with simultaneously.

NoCs are modular because they are built with only two parameterisable components (routers and network interfaces), that are combined in a scalable fashion to form the complete interconnect. New IP blocks can easily be added without changing the existing ones and guaranteed communication services assure that the performance of an IP block is not affected by the performance of other IP blocks. To guarantee bandwidth and latency, resources such as buffers and links must be allocated to connections [4], as we shall see later. The use of an automated tool chain that generates and verifies NoC hardware and software [6] is a key ingredient for successful deployment of NoCs.

Considering the analysis above the introduction of NoCs is unavoidable and the question becomes ”What is the impact on area and performance?”. This isn’t easy to quantify. In [12] a general (artificial) design example is used. This paper follows a different approach. We start from two real-life applications and use two bus-based reference designs, one for an audio application and one for a video application. Audio and video applications have different demands in terms of communication, i.e. the required communication bandwidth and burst size for video are larger than for audio. The reference design for audio is NXPs in-car digital radio [18, 2]. The reference design for video is a programmable multi-standard Orthogonal Frequency-Division Multiplexing (OFDM) receiver [11, 9]. We compare the existing bus-based reference system-on-chips and compare it with several alternative NoC-based solutions.

The outline of this paper is as follows. Section 2 describes the in-car digital radio solution which is used as the reference design for our audio application. Section 3 describes the multi-standard OFDM demodulator and decoder which is used as the reference design for our video application. The NoC architecture is introduced in Section 4. Section 5 evaluates different NoC designs and compares these with the two reference designs. Finally, in Section 6 conclusions are drawn.

2 In-Car digital entertainment

In this section we introduce the reference design for our audio application and extract the application communication requirements. These requirements are used for dimensioning the NoCs, which eventually will be compared with the interconnect of the reference design.

The reference design is NXP’s in-car digital radio chip SAF7780 [18, 2]. The SAF7780 is among others things capable of terrestrial reception, compressed audio playback and handsfree voice with acoustic echo cancellation, possibly in different use-cases like single versus dual media sound. Next to the audio application, the user application is

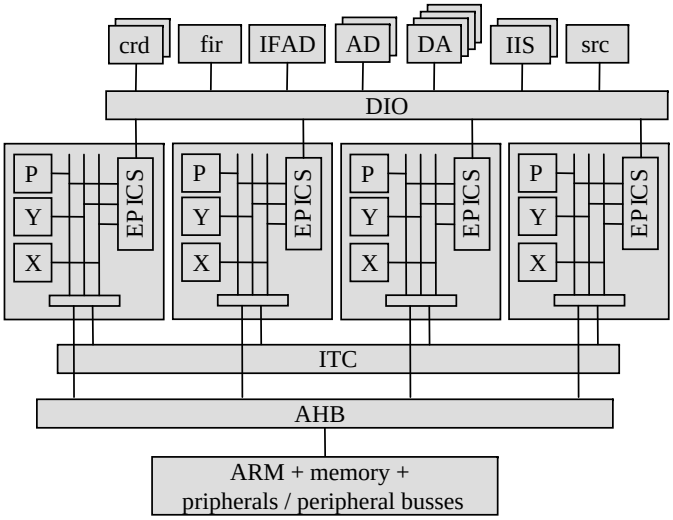

Fig. 1. The architecture of the SAF7780

executed on a programmable CPU which is integrated in the chip. In this paper our focus is on the audio application.

2.1 SAF7780 reference architecture

The SAF7780 reference architecture is shown in Fig. 1. It is a heterogeneous multiprocessor architecture combining a programmable CPU core (ARM), programmable DSP cores (EPICS), hardware accelerators (FIR, CRD) and peripherals. There are different interconnects in different parts of the architecture. The main interconnects are the Inter Tile Communication (ITC), the Digital In/Out (DIO) switch and a multilayer Amba High-speed Bus (AHB). An ARM subsystem, connected to this AHB, is used to configure and bootstrap the chip. Part of the user application is also executed on this ARM processor.

The four EPICs cores together with the ITC and DIO interconnects are the DSP subsystem where most of the signal processing takes place, e.g. audio processing. In our comparison we focus on this subsystem and replace the ITC and DIO interconnects with a NoC. The ITC channels and DIO switch are briefly described below:

ITC channels: an EPICS DSP core with its local memory (P, Y and X) is called a tile. An EPICS DPS core can write data in the memory of another tile, via an ITC channel. Reading from the memory of another tile is not implemented because it was not required by the application. There is an ITC channel from every tile to all other tiles. Tiles and the ITC interconnect are clocked at 125MHz.

ITC is based on address-based transactions. Each tile has its own address space. A specific region of this address space is mapped to an ITC channel. The ITC channel translates the address coming from the source tile to the address in the address space

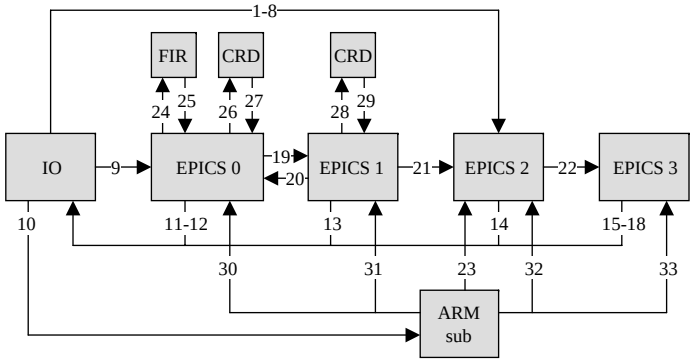

Fig. 2. Application requirements after mapping the audio application

of the destination tile. The address ranges and address translations are programmable at run time. Typically they are programmed only once per use case (mode).

DIO switch: The DIO switch connects four EPICS DSP cores to peripherals and application specific cores (hardware accelerators). Registers in peripherals and application specific cores are memory mapped in the address space of an EPICS DSP core. The peripherals, application specific cores, DIO switch and tiles are synchronous, clocked at 125MHz.

Each peripheral and application specific core is assigned to only one EPICS DSP core so that no arbitration is needed, therefore, the EPICS DSP core can access the data in one clock cycle. The assignment of peripherals and application specific cores to EPICS DSP cores is programmable at run time. This assignment is programmed only once per use-case (mode).

2.2 Communication requirements

The audio application processes streams of data and has real-time constraints. Such a streaming application can be presented by a graph that consists of tasks that communicate via channels, which are mapped onto the interconnect. The application communication requirements as a number of such connections is shown in Fig. 2.

Connections 1 through 18 have a peripheral as a source or destination. In Fig. 2 these peripherals are represented by the Input/Output (IO) box to keep the figure simple. In the NoC-based architectures each peripheral is connected to a network interface port.

The connections are data connections but there are also control connections. So there are two traffic classes:

- *Data connections (1-29)*: streaming connections represented by the edges in the task graph. Symbol sizes vary between 1 word and 512 words.
- *Programming connections (30-33)*: are used only at application start up to load the program memories and control registers of the various cores and IP blocks.

The Bandwidth and latency requirements are as follows:

Bandwidth requirements: the connection bandwidth requirements are derived from the overall symbol throughput and symbol sizes. The symbol throughput is 8 Khz for speech (telephone and navigation), between 40 and 48 Khz for audio and 325 KHz for the modulated radio signal. Symbol sizes vary from 1 word for a mono sample to 512 words for the input of the MP3 decoder.

The audio application has low average bandwidth requirements and most of the communication bursts are small. The average bandwidth requirements for connections 1-23 is between 40 KBytes/sec for the MP3 decoder input and 2 MByte/sec for the terrestrial radio demodulation input. The required average bandwidth between an EPICS DSP core and the Coordinate Rotation Digital computer (CRD) hardware accelerator is approximately 44 MByte/sec, which can be accommodated easily by the interconnect.

The amount of bandwidth assigned to programming connections effects only the stat up time of the application, which is not critical. Therefore, little bandwidth is given to these connections.

Latency requirements: Latency influences both (i) total time data takes to pass through the processing chain, and (ii) the throughput if the graph contains loops due to feedback or control. The loops cause a problem because they limit the possibility of pipelining and algorithmic transformations are needed to increase the performance.

The SAF7780 contains an adaptive filter with such a feedback loop. New filter coefficients are calculated and updated for every sample. The calculation of the filter coefficients is computed on the EPICS DSP core in cooperation with the CRD hardware accelerator. The round-trip latency, from the DSP to the accelerator and back, is composed of interconnect latency and computation latency. Backward compatibility of software is possible if the round-trip latency is not increased, after replacing the DIO switch with a NoC.

The SAF7780 is implemented in 0.18 μm technology. The EPICS DSP core and CRD hardware accelerator share the same clock with a clock frequency of 125 MHz. The EPICS accesses the input and output registers of the CRD in one clock cycle. The round-trip latency is determined by the computation latency of the task executed on the CRD, which is 36 clock cycles. Therefore, the round-trip latency is $36/(125 \cdot 10^6) = 288$ ns.

3 Digital Video Broadcasting - Terrestrial

In this section we introduce a second reference architecture which is a demonstrator and prototype of a fully programmable multi-standard OFDM demodulator and decoder using Silicon Hive cores. It is therefore a true software-defined radio design. In this paper we focus on the Digital Video Broadcasting - Terrestrial (DVB-T) [15, 3] application.

3.1 OFDM reference architecture

The OFDM reference architecture is shown in Fig. 3. It includes the processing cores (Bresca, Avispa1, Avispa2, Fec Inner, Fec Viterbi, Fec Outer) and peripherals with their

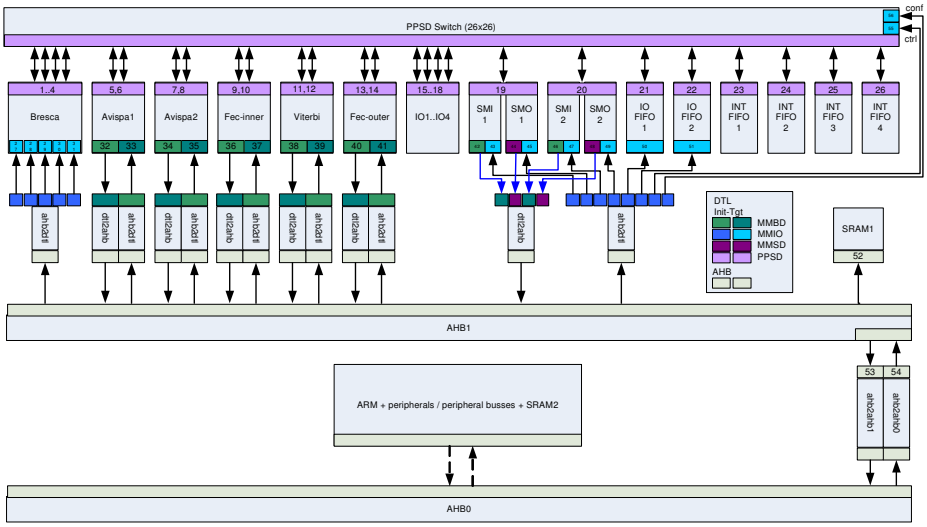

Fig. 3. Overview of the reference OFDM architecture. Note the AHB2DTL blocks that convert DTL to AHB and vice versa.

interconnects. The main interconnect structure is a bridged multilayer Amba High-speed Bus (AHB0 and AHB1) and a semi-static Peer-to-Peer Streaming Data (PPSD) switch. An ARM subsystem, connected to AHB0, is used to configure and bootstrap the processing cores. Most of the IP components use Philips's Device Transaction Level protocol (DTL) [14] as the interconnect-independent interface. The DTL is based on 4 profiles that support address-less streaming (PPSD) and single/burst/stream address-based transactions (MMIO/MMBD/MMSD, respectively). Adapters are used to convert from DTL to interconnect-specific protocols such as AHB and back. Notice that some adapter blocks also function as concentrators/distributors multiplexing bus traffic to/from multiple IP ports.

In our comparison we replaced AHB1 and the PPSD-switch with a NoC as these constitute the critical communication subsystem. The multilayer AHB and PPSD-switch are briefly described below:

Multilayer AHB: the Amba High-Speed Bus (AHB) [1] is a high-speed bus architecture. Multi-layer AHB (ML-AHB) and AHB-lite are super- and subsets, respectively, of this architecture. AHB-lite is a subset of the AHB bus protocol which only allows for one master, requiring no arbitration and saving some signals (request, grant, retry and split).

Multi-layer AHB (ML-AHB) is an interconnection architecture that extends the AHB bus architecture. It provides parallel accesses between multiple masters and slaves (Fig. 4) to increase the overall bus bandwidth and flexibility in the system architecture. The ML-AHB crossbar interconnection matrix has a higher area cost than standard AHB. The number of bus layers in one bus segment depends on performance and

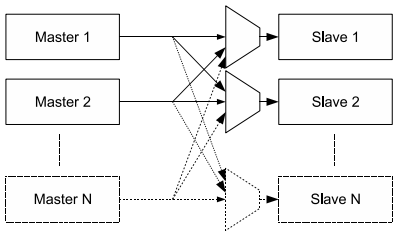

Fig. 4. Schematic of master-to-slave paths of a N-layer AHB(-lite) system.

clock-speed constraints (due to layout/placement). To achieve high clock speeds it may be desirable to split the bus.

The ML-AHB1 bus segment we take in our comparison is designed and verified for 80 MHz operation and contains 8 AHB-lite layers, each layer providing full connectivity to all slaves.

PPSD Switch: part of the interconnect structure is based on streaming point-to-point channels (DTL-PPSD). The PPSD switch allows connections to be programmed at run time. It consists of a single crossbar switch implemented using multiplexers and input/output FIFOs, and is clocked at 80 MHz. Connections are point to point and set up only once per use case (mode).

3.2 Communication requirements

Fig. 5 displays the DVB-T application communication requirements as a number of connections. Strictly speaking, the concept of connections does not exist for the original architecture because from the processor perspective, communication is address-based and the system is fully connected, i.e. each device can address any other device in the system.

The connections are categorised into four traffic classes:

– *Data connections (1-9)*: high-bandwidth streaming connections. Symbol sizes vary between 1 word and 8K words and are constant per connection.

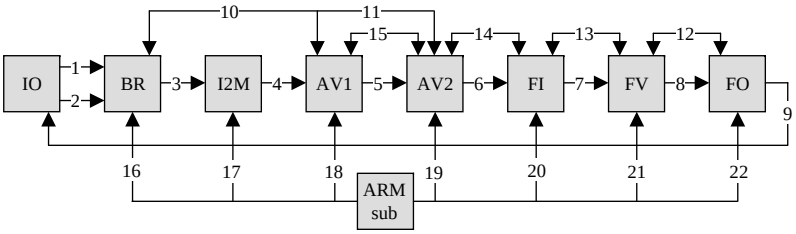

Fig. 5. Application requirements after mapping on cores.

- *Control connections (10-11)*: low-bandwidth streaming connections on which a control word is sent for every DVB-T symbol that a core has processed.
- *Token connections (12-15):* low-bandwidth streaming connections used to send synchronisation tokens between processing cores. Synchronisation is based on available memory blocks.
- *Programming connections (16-22):* are used only at application start up to load the program memories and control registers of the various cores and IP blocks.

The Bandwidth and latency requirements are as follows:

Bandwidth requirements: the connection bandwidth requirements are derived from system's overall required symbol throughput, symbol sizes and processor IO-rates. The DCT processing symbol sizes (8K words), and correspondingly the communication bursts, are large. Moreover, the processors operate a frequency higher than the bus frequency. As a result, the processors can saturate the bus. Hence the peak throughput of the high-bandwidth connections (520 MBytes/sec per connection) is limited by the AHB bus (320 MBytes/sec). The peak throughput is therefore spread out over time. This is allowed because the average throughput per connection ranges from 3 to 36 MWords/sec, which can be accommodated easily by the interconnect. The same reasoning applies to the programming connections. They are given little bandwidth because it affects only the start-up time of the application, which is not critical.

Latency requirements: the interconnect latency is negligible compared to the total processing time, making control and synchronization connections most latency critical. Cores send a few control tokens to the predecessor core halfway during the processing of a symbol, which should arrive before the next symbol is processed on the predecessor core. However, control loops are present only on cores that process relatively big symbols and, as a result, control latency requirements are low.

3.3 Area cost

Table 1 shows the area cost of the total interconnect. The busses and PPSD switch achieve 80 MHz after synthesis. The total interconnect area amounts to only a few percent of the total system-on-chip area.

In last two sections we introduced two reference designs and their application communication requirements. The NoC, which is described in next section, is dimensioned

Table 1. Interconnect area of the original design.

AHB routing logic	0.119 mm^2
AHB/DTL adapter	0.996 mm^2
PPSD switch	0.563 mm^2
Total interconnect area	1.68 mm^2

with the communication requirements and compared with the interconnects of the reference designs in Section 5.

4 Æthereal NoC

In this section we introduce the relevant characteristics of the Æthereal NoC [7], in particular the network interface (NI) [16].

4.1 NoC architecture

The NoC is composed of NIs and routers interconnected by links. NIs translate the IP protocols to NoC-internal packet-based protocols, offering two types of connections (or service classes): *guaranteed throughput* (GT), and *best effort* (BE). Data that is sent on BE connections is guaranteed to arrive at the destination, but without minimum bandwidth and maximum latency bounds. End-to-end flow control is used to ensure loss-less data transfer. GT connections use time-division multiple access (TDMA) to give hard (worst-case) guarantees on minimum bandwidth and maximum latency. Both GT and BE connections use source routing, i.e. the path to the destination is decided at the initiator NI. The initiator NI must be configured with this path, as we shall see later.

Data is sent from one NI to another using packets and is buffered using wormhole routing for low buffering costs. Every router contains GT input buffers consisting of one flit (3 words of 32 bits), and BE buffers of eight flits (24 words). TDMA router buffers require only one flit, as GT packets never stall in the router network. This is accomplished by globally scheduling packet injection from the NIs to the routers in such a way that packets never use the same link at the same time (thus avoiding contention). The pipelined virtual circuits that are implemented this way have a guaranteed minimum bandwidth (roughly, the number of slots reserved for the GT connection) and bounded latency (roughly, the waiting time until the appropriate slot, plus three cycles per router along the path). The TDMA slot allocation is an optimisation problem, per use case (or mode) of the NoC. We currently solve it at design time, resulting in a number of configurations. At run time these configurations are programmed (or loaded) in the NoC.

BE connections use slots that have not been reserved, or have not been used by GT packets. BE packets are scheduled dynamically at run time, and their behaviour (bandwidth, latency) is therefore not predictable.

4.2 Network interfaces

The network interface (Fig. 6) is split in a fixed *kernel* and variable *shells*. A NI shell converts transactions (e.g. read and write) of a particular IP protocols, such as DTL [14], to transport-layer messages. The NI kernel converts these generic messages into network-layer GT or BE packets. Shells are a modular layered approach: they confine protocol specific functionality; they can be composed to build complex protocols; and they allow multiple different IP ports to use a single NI [16].

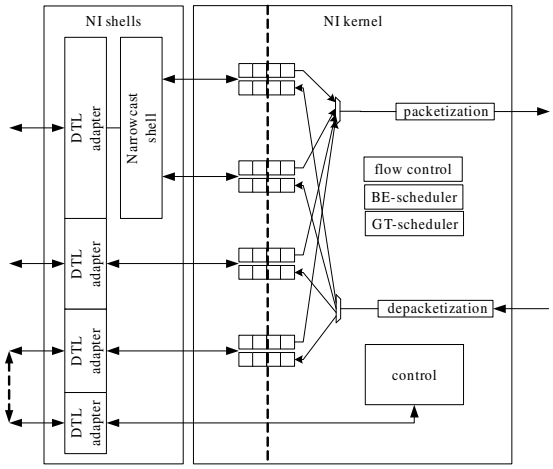

Fig. 6. Simplified network interface architecture.

The NI kernel contains FIFOs for three purposes. (i) They implement the clock boundary between IP blocks and the NoC. (ii) They decouple and isolate IP communication behaviour from the NoC behaviour. That is, data bursts from IP are buffered to fit the TDMA transmission schedule, and vice versa. (iii) They hide the round-trip latency of end-to-end flow control credits, which increases the buffer sizes by a few percent.

A connection uses two channels for every master-slave pair, with two buffers each. In order of use the connection buffers are: the initiator NI request buffer, the target NI request buffer, the target NI response buffer, and the initiator NI response buffer. As an example, Fig. 6 shows an initiator NI with three connections. The bottom two connections are simple connections of a master to a single slave, each using a request and a response buffer. The top connection is a narrowcast connection, in which a master communicates with multiple (in this case two) slaves using two channels. The connection uses four buffers in the initiator NI. In traditional address-based interconnects the processor can address each device. The narrowcast shells transparently implement the address-to-connection conversion for backward compatibility.

NIs must be programmed with the appropriate configuration at run time. This is performed using a memory-mapped IO (MMIO DTL profile [14]) configuration port on each NI. This configuration port is looped back to a target IP port (the bottom port in Fig. 6). The NoC is configured using itself, and no separate control interconnect is required [7].

4.3 NoC design flow

The NoC design flow we use consists of a number of tools for NoC generation, IP mapping, configuration, performance verification and simulation as shown in Fig. 7. Tools communicate using XML formats [6]. Note that our experiments do not include recent improvements to mapping, routing, and TDMA slot and buffer allocation [10].

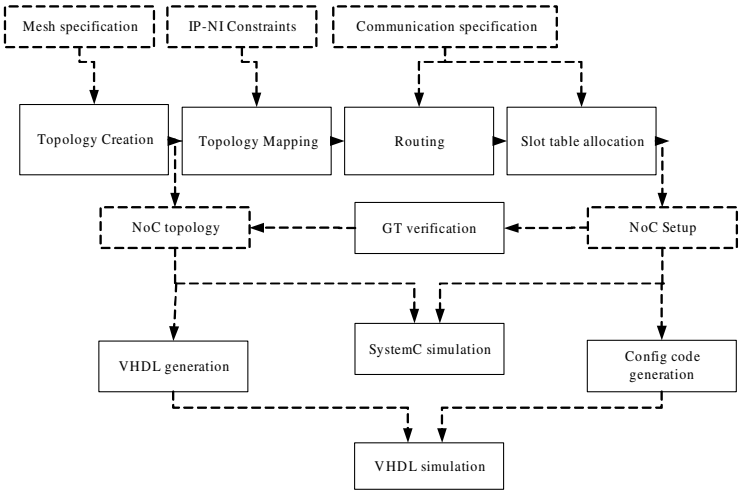

Fig. 7. NoC design flow overview.

The input to the NoC flow consists of the specification of the required communications (i.e. connections) for each use case (mode). For each connection the required protocol, bandwidth, latency, and burst size is specified. For all ports on all IP blocks the protocol and protocol related settings are also given. First, the NoC topology is selected, and the mapping of IP ports on the NI ports is determined. The topology XML file, with back-annotated buffer sizes, is used to generate RTL VHDL for gate-level synthesis.

The routes through the NoC of all connections, and the TDMA slots of all GT connections are then computed. The resulting XML file can be used to configure the NoC directly, or can be translated to C for compilation on embedded processors that configure the NoC.

The NoC description, the IP port mapping, and the configuration are used by the GT verification tool, which analytically verifies the guaranteed performance of GT connections, i.e. minimum bandwidth, maximum latency, and required buffer sizes [4].

4.4 NoC Area Cost

The NoC cell area is composed of router area and NI area. The router area depends on the number of routers, their degree (number of inputs and outputs) and the type of router (GT+BE, GT only). The GT and BE buffers in the router have a fixed size. The number of routers and their degree is determined by the topology. We select the smallest topology for which a successful mapping and configuration can be found from a set of templates (meshes in this case). There are two types of routers, GT+BE and GT-only. The GT+BE router contains GT and BE buffers whereas the GT-only router contains only GT buffers. Therefore, the area cost of a GT-only router is smaller than a GT+BE router. For example, a 6x6 GT+BE router occupies 0.175 mm^2, and a 6x6 GT-only router 0.033 mm^2 [7].

The NI area depends on the number of network interfaces, the number of connections and the NI buffers. As mentioned before, the buffers decouple the IP behaviour from the network behaviour and vice versa. A larger transaction burst size means more bursty traffic, and a larger buffer is required to decouple the IP and the network. The buffers must also hide the round-trip latency of end-to-end flow control credits. The size of NI buffers therefore depends on the connection's transaction burst size and round-trip latency, which in turn depend on the NoC topology, the mapping of IP ports to NI ports, the routing, the number of slots in the TDMA table, and the TDMA slot allocation. These parameters are mutually dependent.

The TDMA table size and slot allocation are determined by the usage of the NoC links. TDMA serves two purposes: to allocate and enforce different bandwidths to different connections, and to avoid contention (described before). Contention occurs within the router network, but also at the links between routers and NIs. Especially the latter depends very much on the mapping of IP ports to NI ports: if many connections use the same NI-router link a large TDMA table is required. The former depends mostly on the topology. A star topology, for example, funnels all connections to a single bottleneck, and requires a large TDMA table. A highly connected topology has less contention because links are less used, and because alternative paths may be available to route around congested areas.

Thus, the TDMA table size, and the slot allocation are determined by the quality of the mapping, routing, and TDMA slot allocation algorithms. We use XY routing, with an incremental slot allocation algorithm. IP port to NI port mapping balances IP port bandwidths over the NIs, clustering IP ports that communicate heavily on the same NI. It then minimises the distance (number of hops) between heavily communicating NIs, taking care not to overload any link. The improved UMARS algorithm [10] that reduces the TDMA table size, and improves the slot allocation for small buffers was not yet available at the time of our experiments for the ODFM case study. The improves UMARS algorithm is used for the SAF7780 case study.

Reducing the area of the NoC requires a trade off between minimising the number of routers and NIs, and minimising contention (which is easier in a larger NoC).

Assuming 500 MHz operation, testable, with worst-case military back-annotated lay-out timing, in Philips's 0.13 μm process technology, with GT+BE routers, [5] determined the following estimations for the router (Equation 1) and NI (Equation 2) area, respectively. In Equation 1 and Equation 2, p denotes the number of ports, c the number of connections per port, q the average buffer depth, and a the router degree. For the NI and router buffers, Equation 1 and Equation 2 count for hardware ripple-through FIFOs [19] which are faster and smaller than flip-flop-based FIFOs.

$$A_{\mathrm{R}}(a) = (0.808a^2 + 23a) \cdot 10^{-3}\ \mathrm{mm}^2 \tag{1}$$

$$A_{\mathrm{NI}}(p, c, q) = (19.6pc + 0.72pcq + 4.8) \cdot 10^{-3}\ \mathrm{mm}^2 \tag{2}$$

The designs we compare with in Section 5.1 are all based on minimal TDMA tables. Furthermore the number of NIs connected to the processing cores and their mapping was chosen largely equal to the traditional interconnect structure (e.g. each processing core has a single NI connected to it) as to facilitate the comparison. The tools assume operation of the NoC at 500 MHz.

4.5 Connection latency

Latency on a connection from source to destination is defined as the difference between the time at which the first word of a message has been offered to the network and the time the last word has been delivered to the destination. That is the so-called *total latency*. Total latency is composed of the *waiting latency* and *network latency*. If the NI has no space to accept data coming from the IP, the IP is stalled.

Waiting latency is defined as the difference in time at which the first word of a message has been written in the initiator NI request buffer and the time this word has been scheduled for packetization. Two factors contribute to this latency, (i) the first word of a message has to wait until it reaches the head of the buffer and (ii) it has to wait until it is scheduled. The latter depends on the distance between two allocated slots in the TDMA table.

Network latency is a consequence of latency in the NI shells, NI kernels, clock domain crossings, routers, arbitration and end-to-end flow control. The NI shell introduces two cycles latency in our DTL master shell (due to sequentialization, as part of packetization) and zero to two cycles latency in the narrowcast and multicast shells (depending on the NI instance). Between one and three cycles latency is introduced by the NI kernels (as data needs to be aligned to a three word flit boundary). The clock domain crossing, between source and destination, introduces two clock cycles latency at the destination clock. Three clock cycles latency is introduced per router. The TDMA arbiter causes additional delay. Per TDMA wheel rotation a predefined number of words can be send over the network. Therefore, the additional latency is a predefined number of TDMA wheel rotations, which are necessary for transferring the message. The round-trip latency of end-to-end flow control credits is hidden by sufficiently large NI buffers [4].

Note that a messages sent on a BE connection has an unbounded latency, because in the TDMA table no slots are allocated to this connection. From the messages sent on GT connections it is possible to compute an upper bound from a multi-rate dataflow model of the GT connection [13].

5 Comparison

In this section we compare the traditional interconnect of each of the two reference designs with replacement Æthereal NoCs. The area cost of the NoC is computed for various NoC configurations. The latency of the critical connections is computed for one NoC configuration.

5.1 Area comparison

In this subsection we assess the impact of (i) the NoC topology, (ii) the use of GT+BE versus GT-only routers, (iii) the number of connections in the design, and (iv) buffer depth optimisation.

(i) NoC topology: to explore the impact of the topology on the NoC area we implemented different NoCs without further optimisations. Table 2 contains the estimated

Table 2. Effects of topology scaling

Design	saf7780_1	saf7780_2	ofdm_1	ofdm_2	ofdm_3	ofdm_4
Mesh	1x1	1x2	1x1	1x2	2x2	3x3
# NIs	8	8	8	8	8	9
# TDMA slots	11	11	3	8	8	5
# buffers	148	148	132	132	132	134
Avg. buffer size (words)	6.23	6.52	8.81	9.30	9.42	9.19
FF-FIFO-based synth. (mm^2)	-	-	4.16	4.43	4.84	5.98
Opt.-FIFO-based synth. est. (mm^2)	-	-	1.79	1.84	1.95	2.33
Opt.-FIFO-based est. (mm^2)	2.39	2.45	1.99	2.04	2.20	2.66

area results. For synthesis of the NoC we use the Synopsys Ultra Design Compiler using Philips's 0.13μm technology and the same wire-load model as the OFDM reference design. Synthesis effort was set to medium with 200 MHz target clock speed.[6] For the NI and router buffers, the NoC designs used either synthesisable flip-flop-based FIFOs (FF-FIFO-based), or estimated area for faster and smaller hardware ripple-through FIFOs [19], referred to as "optimised FIFOs" (Opt.-FIFO-based). The rows labelled "FF-FIFO-based synth." and "Opt.-FIFO-based synth. est." contain the NoC area as obtained by synthesis of the entire NoC using FF-based FIFOs, and a synthesis based estimate using optimised FIFOs, respectively. The row labelled "Opt.-FIFO-based est." shows the estimate made by the automated tool chain, using Equation 1 and Equation 2.

The saf7780_1 and saf7780_2 designs are based on 29 GT and 4 BE connections. Most of the connections have low bandwidth requirements. BE connections are used for programming connections. All connections can be programmed at run time to be either GT or BE. The configuration processor uses only one connection to program the NoC. The buffer sizes of the GT connections were computed by the NoC design flow and for the BE connections buffers of sizes 8 words where used.

The ofdm_1 - ofdm_4 designs are based on 9 GT and 7 BE connections. They contain 13 additional zero-bandwidth (ZB) connections. The ZB connections are not used in the DVB-T application, but provide connectivity for other OFDM-based use cases. The NoC therefore offers the same connectivity as the traditional interconnect, for a correct comparison. GT connections are high bandwidth and used for the data flow of the application. BE connections are used for the low-bandwidth tokens, i.e. control and programming data. For the BE and ZB connections buffers of size 8 words were used, supporting only low bandwidth communication. The buffer sizes of the GT connections were computed by the NoC design flow.

saf7780_1-saf7780_2 and ofdm_1-ofdm_4 are all meshes, but of different sizes. Recall that the TDMA table size is affected by the number of connections sharing links between NIs and router (depending mainly on the mapping), and the contention on links (depending on slot allocation and routing). The 1x1 meshes only have NI-router contention, leading to a TDMA table with 3 and 11 slots for ofdm_1 and saf7780_1, re-

[6] An unoptimised narrowcast shell limited the NoC speed to 200 MHz, all other parts of the design reached higher clock speeds.

spectively. The 1x2 (ofdm_2, saf7780_2) and 2x2 (ofdm_3) meshes additionally suffer from contention in the NoC. The (heuristic) mapping, routing, and slot allocation cannot compensate for this, and 8 slots are required for the OFDM designs. In saf7780_2 the TDMA table contains still 11 slots because of the NI-router contentions. The 3x3 mesh (ofdm_4) offers more freedom to the algorithms, reducing the TDMA table to 5 slots. Although the TDMA table size impacts the NI buffering cost of high-bandwidth connections, the large number of low-bandwidth connections lowers the impact on the NI area. The difference in router area has the most impact on the total NoC area.

Table 3. Area estimation of GT-only optimisation

Design	saf7780_1gt	saf7780_2gt	ofdm_1gt	ofdm_2gt	ofdm_3gt	ofdm_4gt
Mesh	1x1	1x2	1x1	1x2	2x2	3x3
# NIs	8	8	8	8	8	9
# TDMA slots	12	11	9	17	17	11
# buffers	148	148	132	132	132	134
Avg. buffer size (words)	7.42	7.02	9.86	11.29	11.33	10.24
Opt.-FIFO-based est. (mm^2)	2.32	2.30	1.85	1.93	2.00	2.15
Difference with BE+GT	-2.9 %	-6.1 %	-7.0 %	-5.4 %	-9.1 %	-19 %

(ii) GT+BE versus GT-only routers: the area of the NoC can be reduced by using GT-only routers, because the 6x6 GT+BE router occupies 0.175 mm^2, and a 6x6 GT-only router 0.033 mm^2 [7]. The row labelled "Opt.-FIFO-based est." of Table 3 contains the estimated NoC area with optimised hardware FIFOs and smaller GT-only routers. All BE connections are converted to GT connections. As a result, the size of the TDMA table increases to accommodate the additional connections. All buffer sizes are now computed by the NoC design flow, formerly for BE connections buffer sizes of 8 words where used. The average channel buffer sizes grow for both designs. Of course, the former BE connections now have a guaranteed throughput.

The previous designs demonstrate that the NoC cost is mainly determined by the number of connections (i.e. number of buffers) and the TDMA contention in the NoC (affecting the TDMA table and the sizes of the buffers). We have illustrated how a larger NoC (more routers) reduces TDMA contention (and hence buffer cost), with ofdm_1-ofdm_4. Larger NoCs approximate a fully connected switch with least TDMA contention (i.e. one router, which is not scalable). We also illustrate that converting BE connections to GT connections reduces the router area at the cost of increased TDMA contention (ofdm_1-ofdm_3 versus ofdm_1gt-ofdm_3gt and saf7780_1-saf7780_2 versus saf7780_1gt-saf7780_2gt).

Table 4. Area results after connection optimisation

Design	saf7780_2	saf7780_2b	ofdm_3	ofdm_3b	ofdm_3c	ofdm_3d
Mesh	1x2	1x2	2x2	2x2	2x2	2x2
# NIs	8	8	8	8	8	8
# TDMA slots	11	8	8	8	6	6
# buffers	148	104	132	52	52	52
Avg. buffer size (words)	6.52	7.15	9.42	9.23	7.46	5.58
FF-FIFO-based synth. (mm^2)	-	-	4.84	2.50	2.30	2.12
Opt.-FIFO-based synth. est. (mm^2)	-	-	1.95	0.98	0.94	0.90
Opt.-FIFO-based est. (mm^2)	2.45	1.86	2.20	1.14	1.11	1.07

(iii) number of connections: the following designs use specific optimisations that are design dependent, unlike the previous trade offs that could all be automatically generated by the design flow.

In saf7780_2b the number of GT connections is reduced from 29 to 18 for determining the impact on the number of connections. This reduction of connections in the NoC is achieved by sharing the low-bandwidth connections from and to peripherals by means of combining the peripherals in one tile. The number of buffers is reduced from 148 to 104. Although, the number of TDMA slots are reduced (11 to 8) the average buffer depth is slightly increased (6.52 to 7.15) because we remove mainly small buffers. When comparing the area of saf7780_2b with the traditional interconnect, the increase is a few percent on the total chip area.

ofdm_3b is based on the 9 high-bandwidth GT connections only, and unused ports are removed. This resembles the application's main data flow only. ofdm_3b serves to assess the impact of the low-bandwidth connections on the NoC. We remove them from the NoC with the assumption that we can share the low-bandwidth connections (i.e. buffers & TDMA slots). The number of TDMA slots is not lower, but the number of buffers is more than halved (132 to 52). However, the average buffer depth does not change much (9.42 to 9.23). In other words, the low-bandwidth connections (using either GT or BE) use a significant number of buffers, but do not cause much contention.

ofdm_3c further reduces NI buffering by limiting the peak throughput from the application's maximum (520 MBytes/sec), for which the NoC was dimensioned, to the theoretical maximum of the traditional interconnect (320 MBytes/sec). This gives a fairer comparison with the reference interconnect. The size of the TDMA table is reduced (from 8 to 6), as is buffering (9.23 to 7.46 average buffer depth).

(iv) buffer depth optimisation: ofdm_3d takes the previous optimisation one (dangerous) step further. Rather than allocate the maximum (worst-case) throughput, it uses simulation to determine the required buffer sizes. This can be achieved by simulating the entire SoC with infinite buffers and recording their maximum fillings. This reduces the maximum buffer sizes of the high-bandwidth connections from $\sim$50 to $\sim$10. Of course, these maxima result from a limited number of simulations, and may not be large enough to guarantee bandwidth and latency, unlike the analytically computed buffer sizes.

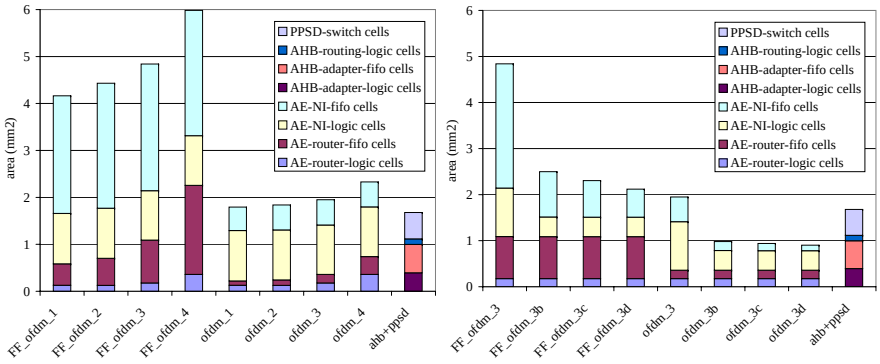

Fig. 8. (Left) Area comparison results for different mesh sizes. (Right) Area comparison results for connection-optimised Æthereal designs. The left-most four designs prefixed with FF are based on flip-flop FIFOs, the next four designs are based on optimised FIFOs. The right-most column contains the original interconnect area break down for the OFDM reference design.

The impact on the average buffer size and total area is small because only the request buffers of (write-only) high-bandwidth connections are reduced. The response buffers and programming connections are not changed. For the saf7780 designs the buffer sizes are already small, therefore, the impact on the average buffer size and total area is limited.

Fig. 8 contains a bar chart with the left-most eight bars showing the area of ofdm_1-ofdm_4 for FF-FIFO-based synth." and "Opt.-FIFO-based synth. est.", respectively. The right-most bar contains the original interconnect breakdown. We divided the area in logic for routing (bus or routers), logic for (bus or network) interfaces, and buffering cost (all buffers and state variables in the interconnect). This distinction can be easily obtained from gate-level synthesis. Buffering cost includes all flip-flops and FIFOs (RAMs are not used).

In this section we investigated the impact of the NoC topology, the use of GT+BE versus GT-only routers, the number of connections in the design, and buffer depth optimisations. Below we investigate the latency of the critical connections.

5.2 Latency comparison

In the OFDM designs control loops are present only on cores that process relatively big symbols and, as a result, there are no stringent latency requirements. However, in the SAF7780 design the audio application contains a control loop which lead to a tight latency constraint in the communication between the EPICS DSP core and the CRD hardware accelerator. This round-trip latency is 288ns with the current DIO switch running at 125 MHz, as described in Section 2.2. In a NoC-based architecture, the EPICS and CRD are attached to two different NIs. Clock domain crossing in the NI kernels enable the CRD to process at a higher clock frequency. The round-trip latency is composed of interconnect latency and computation latency. A higher clock frequency

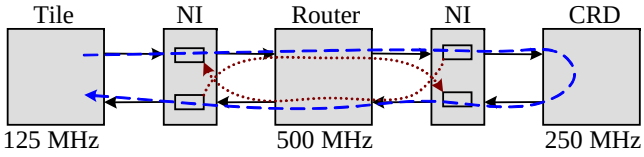

Fig. 9. Round-trip latency between the EPICS DSP core and CRD hardware accelerator

for the CRD results in a lower computation latency and more relaxed constraint for the interconnect latency.

The round-trip latency from the EPICS to the CRD and back to the EPICS is illustrated in Fig. 9 with the dashed arrow. For the purpose of analysing the round-trip latency we assume an implementation in 0.13 μm technology. The clock frequency of the EPICS is taken 125 MHz, which is the same as in the SAF7780. The network can run at a clock frequency of 500 MHz in 0.13 μm technology. In this technology it is expected that the CRD can run at a clock frequency of 250 MHz. There is a connection from the EPICS to CRD and a connection from the CRD to EPICS. Both connections are configured as GT connections. The end-to-end flow control credits of one connection are piggy-backed on messages send over the other connection, as illustrated with the dotted arrows in Fig. 9.

The task executed on the CRD has a computation latency of $36/(250{\cdot}10^6) = 144$ ns. The interconnect latency depends on the length of the message and allocation of slots in the TDMA table. The CRD reads four words from the input connection and writes two words to the output connection. In the case that address-less streaming communication (PPSD) is used, no extra data (e.g. control and address) is send in a message. Low latency results can be achieved by reserving many slots spread over the TDMA wheel, in such a way that the distance between reserved slots is small. For example, reserving one slot out of every two consecutive slots results in a 50% bandwidth allocation and an upper bound on the total NI latency of 108 ns. The latency introduced by the clock domain boundaries and router are 32 ns and 12 ns, respectively. Therefore, the total round-trip latency is 144+108+32+12=296 ns. This round-trip latency is 2.7 % higher than the round-trip latency in the SAF7780 (which is 288 ns) but is still acceptable.

In this section we investigated the impact of the NoC topology, the use of GT+BE versus GT-only routers, the number of connections in the design, and buffer depth optimisations. Finally, we computed the NoC latency for the critical connections in the SAF7780. Below we draw a number of conclusions.

6 Conclusions

In this paper we presented an interconnect comparison based on two existing software defined radio designs, one for in-car radio and one for DVB-T. For these two designs we conclude that it is feasible to replace the traditional interconnects by an Æthereal NoC and still meet the communication requirements (bandwidth and latency).

NoCs offer a structural and scalable approach for the integration of IP blocks into a working system. They help to master the deep sub-micron VLSI design problems by structuring the top level wires in a chip and facilitate modular design. On a software level the guaranteed communication services, offered by the Æthereal NoC [7], are a step forward in mastering the programming effort.

Based on two case-study implementations we conclude that the NoCs are competitive in terms of area with current dedicated interconnects. The NoC designs demonstrate that the NoC area cost is mainly determined by the number of connections (translating to a number of buffers) and the network topology (affecting the number of routers, the TDMA table and the sizes of the buffers).

We have illustrated how a larger NoC with more routers reduces TDMA contention and hence buffer cost. Larger NoCs approximate a fully connected switch with least TDMA contention (i.e. one router, which is not scalable).

The GT-BE trade off (using BE connections and GT+BE routers, or only GT connections and GT-only routers) is valuable, leading to an area reduction of 19% for a 3x3 mesh NoC. Converting BE to GT increases TDMA contention and hence buffer sizes, but this is offset by the lower cost of GT-only routers (0.033 mm^2 instead of 0.175 mm^2 for GT+BE routers).

The large number of low-bandwidth peripheral connections causes most problems. Either they use GT connections and increase TDMA contention (but not too much), or they result in the use of BE connections and (expensive) GT+BE routers. Essentially it is their number rather than their low-bandwidth that causes most cost.

The current Æthereal design flow already automatically finds the smallest regular topology (mesh, etc.), with the smallest TDMA table and optimised FIFO sizes. Optimised ripple-through hardware FIFOs are an essential component of the Æthereal NoC, leading to area reductions of around 60%. The experiments in this paper have shown that it is worthwhile to also automate the BE-GT trade off.

Furthermore, future work will include converting multiple BE connections to a single connection with shared buffers and shared TDMA bandwidth. Although the resulting connections are still BE, it reduces the number of connections and the number of buffers, as well as the TDMA contention and the depth of the buffers. The NoC can also use (inexpensive) GT-only routers.

References

1. ARM. Multi-layer AHB. overview., 2001.
2. H. Bhullar, R. van den Berg, J. Josten, and F. Zegers. Serving digital radio and audio processing requirements with sea-of-dsps for automotive applications the philips way. In *Proc. GSPx Conference*, 2004.

3. European Standard (EN) 300 744 V1.5.1. *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for terrestrial television*.
4. O. P. Gangwal, A. Rădulescu, K. Goossens, S. González Pestana, and E. Rijpkema. Building predictable systems on chip: An analysis of guaranteed communication in the Æthereal network on chip. In P. van der Stok, editor, *Dynamic and Robust Streaming In And Between Connected Consumer-Electronics Devices*, volume 3 of *Philips Research Book Series*, chapter 1, pages 1–36. Springer, 2005.
5. S. González Pestana, E. Rijpkema, A. Rădulescu, K. Goossens, and O. P. Gangwal. Cost-performance trade-offs in networks on chip: A simulation-based approach. In *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 764–769, Washington, DC, USA, Feb. 2004. IEEE Computer Society.
6. K. Goossens, J. Dielissen, O. P. Gangwal, S. González Pestana, A. Rădulescu, and E. Rijpkema. A design flow for application-specific networks on chip with guaranteed performance to accelerate SOC design and verification. In *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, pages 1182–1187, Washington, DC, USA, Mar. 2005. IEEE Computer Society.
7. K. Goossens, J. Dielissen, and A. Rădulescu. The Æthereal network on chip: Concepts, architectures, and implementations. *IEEE Design and Test of Computers*, 22(5):414–421, Sept-Oct 2005.
8. K. Goossens, J. Dielissen, J. van Meerbergen, P. Poplavko, A. Rădulescu, E. Rijpkema, E. Waterlander, and P. Wielage. Guaranteeing the quality of services in networks on chip. In A. Jantsch and H. Tenhunen, editors, *Networks on Chip*, chapter 4, pages 61–82. Kluwer Academic Publishers, Hingham, MA, USA, 2003.
9. P. Gruijters, K. Koch, and G. Burns. Flexible embedded processors for developing multi-standard broadcast receivers. In *Proc. GSPx Conference*, 2004.
10. A. Hansson, K. Goossens, and A. Rădulescu. A unified approach to constrained mapping and routing on network-on-chip architectures. In *Int'l Conf. on Hardware/Software Codesign and System Synthesis (CODES+ISSS)*, pages 75–80, Sept. 2005.
11. I. Held and B. Vandewiele. Avispa ch - embedded communications signal processor for multi-standard digital television. In *Proc. GSPx Conference*, 2006.
12. P. Martin. A comparison of network-on-chip and busses. Technical report, white paper downloadable from the Arteris website (www.arteris.com), 2005.
13. A. Moonen, M. Bekooij, and J. van Meerbergen. Timing analysis model for network based multiprocessor systems. In *Proc. ProRISC, 15th annual Workshop of Circuits, System and Signal Processing*, 2004.
14. Philips Semiconductors. *Device Transaction Level (DTL) Protocol Specification. Version 2.2*, July 2002.
15. U. Reimers. *DVB-The family of international standards for digital video broadcasting*. Springer-Verlag, 2nd edition, 2005.
16. A. Rădulescu, J. Dielissen, S. González Pestana, O. P. Gangwal, E. Rijpkema, P. Wielage, and K. Goossens. An efficient on-chip network interface offering guaranteed services, shared-memory abstraction, and flexible network programming. *IEEE Transactions on CAD of Integrated Circuits and Systems*, 24(1):4–17, Jan. 2005.
17. M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli. Addressing the system-on-a-chip interconnect woes through communication-based design. In *Proc. Design Automation Conference (DAC)*, pages 667–672, June 2001.
18. R. van den Berg and H. Bhullar. Next generation philips digital car radios, based on a sea-of-dsp concept. In *Proc. GSPx Conference*, 2004.
19. P. Wielage, E. J. Marinissen, and C. Wouters. Design and DFT of a high-speed area-efficient embedded asynchronous FIFO. In *Proc. Design, Automation and Test in Europe Conference and Exhibition (DATE)*, 2007.