Comparison of An Æthereal Network on Chip and A Traditional Interconnect for A Multi-Processor DVB-T System on Chip

Chris Bartels*, Jos Huisken†, Kees Goossens‡, Patrick Groeneveld*, Jef van Meerbergen*‡
*Eindhoven University of Technology, †Silicon Hive, ‡Philips Research
c.l.l.bartels@tue.nl, {jos.huisken, kees.goossens, jef.van.meerbergen}@philips.com, patrick@ics.ele.tue.nl

Abstract— **Growing complexity of multiprocessor systems on chip (MP-SoC) requires future communication resources that can only be met by highly scalable architectures. Networks-on-Chip (NoCs) offer this scalability and other advantages like modularity, quality-of-service (QoS), possibly smaller area footprint and lower power dissipation.**

Although many papers describe the advantages of NoCs and describe techniques to apply NoCs on certain application domains, few have actually employed the complete design chain to make a netlist level implementation and area comparison [1], [2]. This paper describes the application of the Æthereal NoC to an existing bus-based MP-SoC design and an area comparison with the original interconnect structure down to netlist level.

I. Introduction

Busses and custom interconnects (point-to-point, crossbar switches) dominate interconnects for current multiprocessor SoCs. However, they do not address predictability, scalability and complexity adequately.

Firstly, *predictability*: the behaviour of many SoCs depends on the combined, interrelated behaviours of the IP blocks and the interconnect. The complete SoC must be simulated to validate its performance. Interconnects that offer guaranteed performance, such as the Æthereal NoC [3] used in this paper, decouple the behaviour of different IP blocks from one another and from the interconnect. As a result, the performance of an IP block is not affected by the performance of other IP blocks, and verification is compositional. To guarantee bandwidth and latency, resources such as buffers and links must be allocated to connections [4], as we shall see later.

Secondly, *scalability*: with each new fabrication process the effective SoC design space grows, leading to increases in IP integration, on-chip communication and clock speeds. At wire level this means increases in the number of wires and their timing constraints. NoCs can alleviate these problems by allowing for better wire structuring [2].

Thirdly, *design complexity*: to lower the effort of designing an optimised interconnect for each SoC, interconnects must be modular and amenable to automation. NoCs are built of two parameterisable components (routers, network interfaces), that are combined in a scalable fashion to form the complete interconnect. The use of an automated tool chain that generates and verifies NoC hardware and software [5] is a key ingredient for successful deployment of NoCs.

The goal of this paper is to give the reader insight in cost/performance aspects of NoCs compared to bus-based interconnects. For this purpose we use a baseband design of a digital video terrestrial receiver (DVB-T) [6], [7]. We compare the existing bus-based DVB-T reference SoC and compare it with several alternative NoC-based solutions.

The original design, architecture and software requirements are described in the following Section II. Section III details the alternative NoC designs, and section IV compares the two approaches.

II. Original architecture

The reference design 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 DVB-T application.

A. OFDM reference architecture

The OFDM reference architecture is shown in Figure 1. It includes the processing cores (Bresca, Avispa1, Avispa2, Fec Inner, Fec Viterbi, Fec Outer) and peripherals with their 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 Transaction Device Level Protocol (DTL) [8] 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.

B. Multilayer AHB

The Amba High-Speed Bus (AHB) [9] 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).

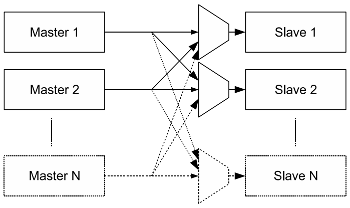

Fig. 2. Schematic of master-to-slave paths of a N-layer AHB(-lite) system.

Multi-layer AHB (ML-AHB) is an interconnection architecture that extends the AHB bus architecture. It provides parallel accesses between multiple masters and slaves (Figure 2) 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 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.

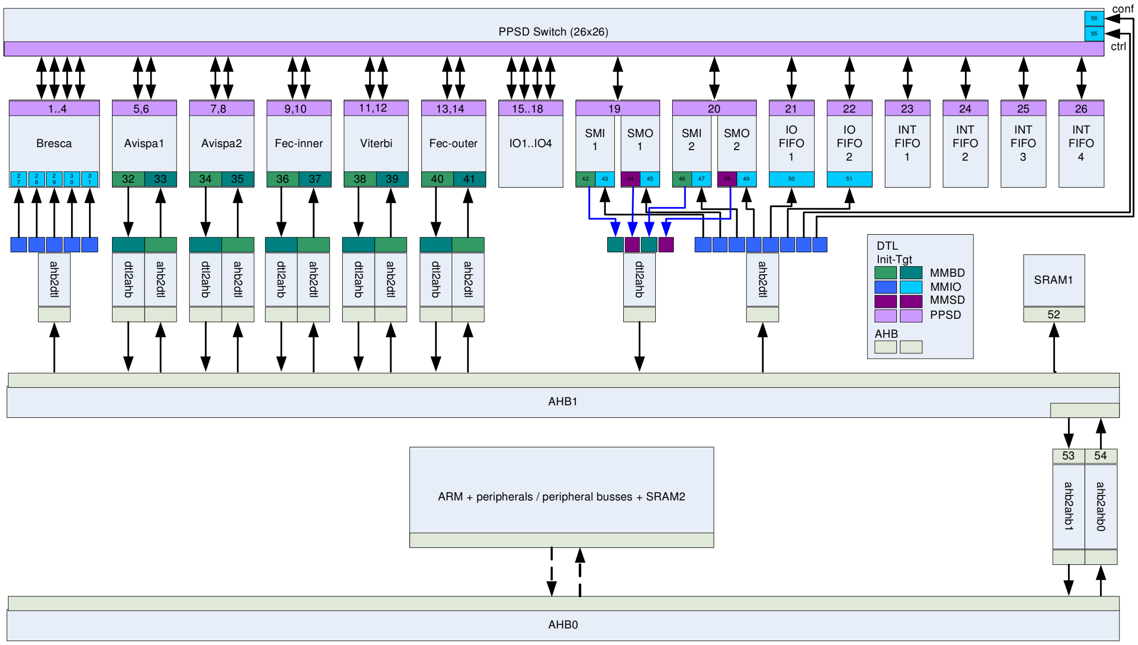

Fig. 1. Overview of the reference OFDM architecture. Note the AHB2DTL blocks that convert DTL to AHB and vice versa.

C. 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 80MHz. Connections are point to point and set up only once per use case (mode).

D. Software IO behaviour

Figure 3 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. In the NoC-based architectures the Æthereal narrowcast shells [10] transparently implement the address-to-connection conversion for backward compatibility.

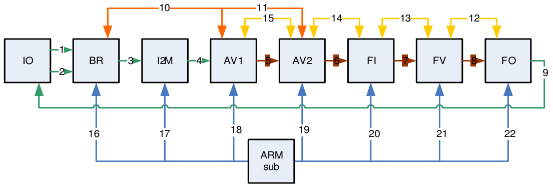

Fig. 3. Application requirements after mapping on cores.

The connections are categorised into four traffic classes:

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

E. 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.

F. Latency requirements

Latency influences both (i) total time data takes to pass through the processing chain, and (ii) the speed of internal control loops, which should not be pipelined for higher performance. The time data resides in the interconnect is negligible compared to the total processing time, making control and synchronisation connections most latency critical. Cores send a few control tokens to the predecessor core half-way 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.

G. Area cost

Table I shows the area cost of the total interconnect. The busses and PPSD switch achieve 80MHz after synthesis. The total interconnect area amounts to only a few percent of the total SoC area.

TABLE I
INTERCONNECT AREA OF THE ORIGINAL DESIGN.

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

III. ÆTHEREAL NOC

In this section we introduce the relevant characteristics of the Æthereal NoC [3], in particular the network interface (NI) [10].

A. 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 latency (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.

B. Network interfaces

The network interface (Figure 4) 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 [8], 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 [10].

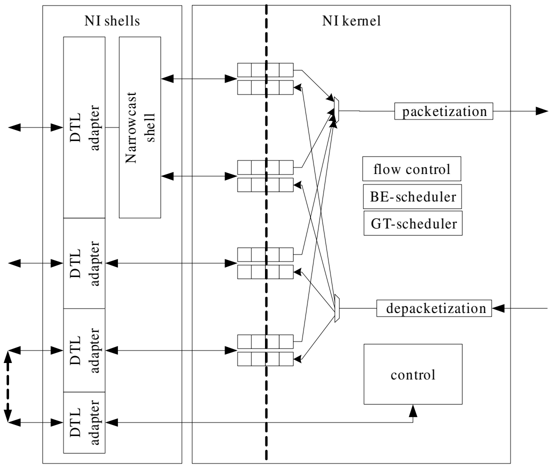

Fig. 4. Simplified network interface architecture.

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, Figure 4 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.

NIs must be programmed with the appropriate configuration at run time. This is performed using a memory-mapped IO (MMIO DTL profile [8]) configuration port on each NI. This configuration port is looped back to a target IP port (the bottom port in Figure 4). The NoC is configured using itself, and no separate control interconnect is required [3].

C. 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 Figure 5. Tools communicate using XML formats [5]. Note that our experiments do not include recent improvements to mapping, routing, and TDMA slot and buffer allocation [11].

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.

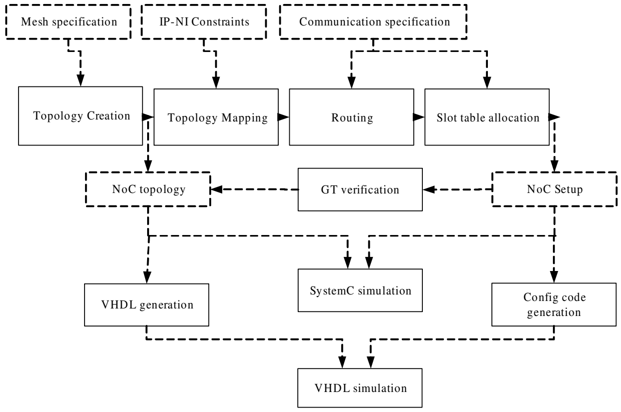

Fig. 5. NoC design flow overview.

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].

D. NoC Area Cost

NoC cell area is composed of router area and NI area. The router area depends only on the number of inputs and outputs (the GT and BE buffers have a fixed size). The NI area depends on the number of channels and the size of their request and response buffers.

Assuming 500MHz operation, testable, with worst-case military back-annotated lay-out timing, in Philips's $0.13\mu m$ process technology, [12] determined the following estimations for the router (Equation 1) and NI (Equation 2) area in $10^{-3}mm^2$, respectively. In (Eq 1) and (Eq 2) p denotes the number of ports, c the number of connections per port, q the average buffer depth, and a the router degree. The NI area comprises parts that are fixed (e.g. schedulers, packetisation), linear with pc (e.g. protocol conversion, fixed part of buffer cost, flow control), and linear in pcq (e.g. variable part of buffer cost, multiplexing logic between port and multiple connections). The router area is made up of a part that is quadratic in the router degree (e.g. core switch), and a part that grows linearly in the degree (e.g. buffering, link level flow control).

$$A_R(a) = 0.808a^2 + 23a \quad (1)$$

$$A_{NI}(p,c,q) = 19.6pc + 0.72pcq + 4.8 \quad (2)$$

As mentioned before, the buffers decouple the IP behaviour from the network behaviour and vice versa. An IP that stays within its declared bandwidth usage will never stall because a buffer is full or empty, once a transaction is started. A larger transaction burst size means burstier traffic, and a larger buffer is required to decouple the IP and the network. The master request and slave response buffers must also hide end-to-end flow control credit latency.

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 [11] that reduces the TDMA table size, and improves the slot allocation for small buffers was not yet available at the time of our experiments.)

The number of routers 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).

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).

The designs we compare with in the following section 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 500MHz.

IV. Area Comparison

In this section we compare the original interconnect and NoCs for the DVB-T SoC. We divide 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-flop and FIFOs (RAMs are not used).

For synthesis we use the Synopsys Ultra Design Compiler using Philips's $0.13\mu m$ technology and the same wire-load model as the reference design. Synthesis effort was set to medium with 200MHz target clock speed.[1] For the NI and router buffers, the NoC designs used either synthesisable flip-flop-based FIFOs, or estimated area for faster and smaller hardware ripple-through FIFOs, referred to as "optimised FIFOs." Eq 1 and Eq 2 use the optimised FIFOs.

The ofdm_1-4 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.

All connections can be programmed at run time to be either GT or BE. 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. The NoC is programmed using only one connection by the configuration processor.

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.

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

TABLE II

COMPARISON OF TOPOLOGY SCALING, GT-ONLY OPTIMISATION, AND NI BUFFER OPTIMISATIONS.

Design	Mesh	# Routers	# NIs	# TDMA	# buffers	Avg. buffer	FF-FIFO-based	Opt.-FIFO-based	Opt.-FIFO-based
		and degree		slots		size (words)	(synth. mm^2)	(synth. est. mm^2)	(est. mm^2)
ofdm_1	1x1	1 8x8	8	3	132	8.81	4.16	1.79	1.99
ofdm_2	1x2	2 5x5	8	8	132	9.30	4.43	1.84	2.04
ofdm_3	2x2	4 4x4	8	8	132	9.42	4.84	1.95	2.20
ofdm_4	3x3	4 3x3, 4 4x4, 1 5x5	9	5	134	9.19	5.98	2.33	2.66
ofdm_1gt	1x1	1 8x8	8	9	132	9.86	-	-	1.85
ofdm_2gt	1x2	2 5x5	8	17	132	11.29	-	-	1.93
ofdm_3gt	2x2	4 4x4	8	17	132	11.33	-	-	2.00
ofdm_4gt	3x3	4 3x3, 4 4x4, 1 5x5	9	11	134	10.24	-	-	2.15
ofdm_3	2x2	4 4x4	8	8	132	9.42	4.84	1.95	2.20
ofdm_3b	2x2	4 4x4	8	8	52	9.23	2.50	0.98	1.14
ofdm_3c	2x2	4 4x4	8	6	52	7.46	2.30	0.94	1.11
ofdm_3d	2x2	4 4x4	8	6	52	5.58	2.12	0.90	1.07

In the following, 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 optimisations.

(i) To explore the impact of the topology on the NoC area we implemented four different NoCs without further optimisations. ofdm_1-4 are all meshes, but of different sizes. Table II and Figure 6 (left) contain the synthesis results. The columns 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 final column labelled "Opt.-FIFO-based est." shows the estimate made by the automated tool chain, using Eq 1 and Eq 2.

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 mesh (ofdm_1) only has NI-router contention, leading to a TDMA table with 3 slots. The 1x2 (ofdm_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. 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 total NoC area.

(ii) The area of the NoC can be reduced by using GT-only routers. The column labelled "Opt.-FIFO-based (est.)" of Table II contains the estimated NoC area with optimised hardware FIFOs and smaller GT-only routers. For example, the 6x6 GT+BE router occupies $0.175mm^2$, and a 6x6 GT-only router $0.033mm^2$ [3]. All BE connections are converted to GT connections. As a result, the size of the TDMA table increases to accommodate the additional (low-bandwidth) connections. This impacts buffering, and the average channel buffer size grows from 9.86 to 11.33. Of course, the former BE connections now have a guaranteed throughput.

The eight 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_1-4. Larger NoCs scalably 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_1-4 versus ofdm_1-4gt.)

(iii) The following two designs use specific optimisations that are design dependent, unlike the previous trade offs that could all be automatically generated by the design flow. Table II and Figure 6 (right) show three optimisations of ofdm_3.

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 GT connections on the NoC. We remove them from the NoC, either by using a smart method to share low-bandwidth GT connections (i.e. buffers & TDMA slots) or by using a separate peripheral bus. 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) 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) 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 ∼50 to ∼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.

The average buffer size and total area effects are less than the reduction in the maximum buffer depths because only the request buffers of (write-only) high-bandwidth connections are reduced. The response buffers and programming connections do not decrease in size.

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 draw a number of conclusions.

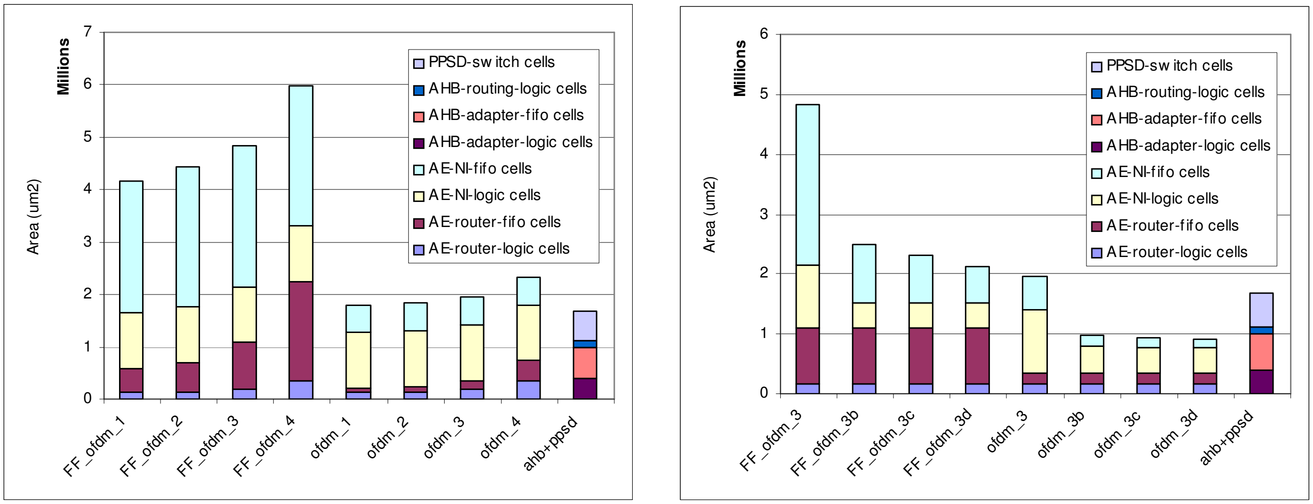

Fig. 6. (Left) Area comparison results for different mesh sizes. (Right) Area comparison results for connection-optimised Æthereal designs. The left-most four designs prefixed with FF are based on flip-flop FIFOs, the next four designs are based on optimised FIFOs. The right-most column contains the original interconnect area break down.

V. Conclusions

In this paper we presented an interconnect comparison based on an existing software-defined radio design for DVB-T. The NoC designs demonstrate that the NoC cost is mainly determined by the number of connections (translating to 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 with more routers reduces TDMA contention and hence buffer cost. Larger NoCs scalably approximate a fully connected switch with least TDMA contention (i.e. one router, which is not scalable). Optimised ripple-through hardware FIFOs are an essential component of the Æthereal NoC, leading to area reductions of around 60%.

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 ($0.033mm^2$ instead of $0.175mm^2$ 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 comparison of the original OFDM interconnect with NoCs is not unfavourable. The ofdm_3gt design with optimised FIFOs implements all connections with their peak throughput, which is more than the original interconnect offers. It only uses general optimisations that could easily be automated. Still, ofdm_3gt is only 16% larger than the interconnect in the original OFDM design, which shows that NoCs are competitive in terms of area with current dedicated interconnects.

The current Æthereal design flow already automatically finds the smallest regular topology (mesh, etc.), with the smallest TDMA table and optimised FIFOs. 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] F. Steenhof, H. Duque, B. Nilsson, K. Goossens, and R. Peset Llopis, “Networks on chips for high-end consumer-electronics TV system architectures,” in *DATE*, Mar. 2006.

[2] F. Angiolini, P. Meloni, S. Carta, L. Benini, and L. Raffo, “Contrasting a NoC and a traditional interconnect fabric with layout awareness,” in *DATE*, Mar. 2006.

[3] K. Goossens, J. Dielissen, and A. Rădulescu, “The Æthereal network on chip: Concepts, architectures, and implementations,” *IEEE Design and Test of Computers*, vol. 22, no. 5, Sept-Oct 2005.

[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 *Dynamic and Robust Streaming In And Between Connected Consumer-Electronics Devices*, Philips Research Book Series, P. van der Stok, Ed. Springer, 2005, vol. 3, ch. 1.

[5] 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 *DATE*, Mar. 2005.

[6] U. Reimers, *DVB-The family of international standards for digital video broadcasting, 2nd ed.* Springer-Verlag, 2005.

[7] *DVB; Framing structure, channel coding and modulation for terrestrial television, European Standard (EN) 300 744 V1.5.1.*

[8] *Device Transaction Level (DTL) Protocol Specification. Version 2.2*, Philips Semiconductors, July 2002.

[9] ARM, “Multi-layer AHB. overview.” 2001.

[10] 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 Trans. on CAD of Int. Circ. and Sys.*, vol. 24, no. 1, Jan. 2005.

[11] 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)*, Sept. 2005.

[12] 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 *DATE*, Feb. 2004.