

# A Hybrid SoC Interconnect with Dynamic TDMA-Based Transaction-Less Buses and On-Chip Networks

Thomas D. Richardson, Chrysostomos Nicopoulos, Dongkook Park,  
Vijaykrishnan Narayanan, Yuan Xie, Chita Das, Vijay Degalahal

*Department of Computer Science and Engineering  
The Pennsylvania State University  
University Park, PA 16802*

*{trichard, nicopoul, dpark, vijay, yuanxie, das, degalaha} @cse.psu.edu*

## Abstract

*The two dominant architectural choices for implementing efficient communication fabrics for SoC's have been transaction-based buses and packet-based Networks-on-Chip (NoC). Both implementations have some inherent disadvantages – the former resulting from poor scalability and the transactional character of their operation, and the latter from inconsistent access times and deterioration of performance at high injection rates. In this paper, we propose a transaction-less, time-division-based bus architecture, which dynamically allocates timeslots on-the-fly – the dTDMA bus. This architecture addresses the contention issues of current bus architectures, while avoiding the multi-hop overhead of NoC's. It is compared to traditional bus architectures and NoC's and shown to outperform both for configurations with fewer than 10 PE's. In order to exploit the advantages of the dTDMA bus for smaller configurations, and the scalability of NoC's, we propose a new hybrid SoC interconnect combining the two, showing significant improvement in both latency and power consumption.*

## 1. Introduction

Current VLSI design trends are shifting toward the System-on-Chip (SoC), or a single die incorporating several homogeneous or heterogeneous processing elements. As the number of processing elements increases, interconnect plays an increasingly major role in SoC design. The type of interconnect used for a specific application will heavily affect the performance and power consumption of the system. A variety of interconnection schemes are currently in use, including crossbars, rings, buses, and networks-on-chip (NoC's) [16]. Of these, the latter two have been dominant in the research community. Buses are a simple, shared-medium interconnect that is quite common and well understood [16]. However, buses suffer from resource contention issues – as the number of processing elements increases, performance

degrades due to excessive conflicts. Furthermore, traditional bus architectures such as AMBA [1] and CoreConnect [13] have inherent disadvantages when applied to SoC's resulting from their transactional model of operation. In those systems, computational elements generally constitute the bus masters and perform read and write operations with slave memory modules or memory-mapped I/O units. Read/write operations are divided into transactions, each of which must be arbitrated, incurring a delay. During high contention periods, this transactional overhead exacerbates access times. To overcome these limitations, attention has shifted toward the other popular interconnect - NoC's. NoC's packetize data and transmit it through an on-chip network. NoC's tout their scalability, just like traditional macro networks, but suffer from variable delay as a result of their multi-hop communication nature, and rapid performance deterioration at high data rates. Furthermore, incorporating more sophisticated routing algorithms and priority schemes dramatically increases the complexity of the routers in NoC systems. This, in turn, incurs significant area, power, and latency overhead, since each router is replicated as many times as there are processing elements.

In this paper, we propose a new transaction-less Time Division Multiple Access (TDMA) bus with dynamic timeslot allocation – the dTDMA bus – in order to circumvent many of the problems associated with traditional bus architectures. In dTDMA, a bus arbiter dynamically grows and shrinks the number of timeslots to match the number of active transmitters. The dynamic nature of this architecture guarantees non-blocking operation, which substantially alleviates the contention problem inherent in bus architectures. This, coupled with its single-hop communication nature and lack of transaction-style arbitration, allows for low and predictable latencies. Hence, dTDMA provides a two-fold advantage over current SoC interconnects: it tackles the contention and transactional issues of existing bus architectures, and avoids the multi-hop overhead of NoC solutions.

We show, in this paper, that these traits allow the dTDMA architecture to outperform traditional bus architectures, as illustrated by an example comparison with the AMBA bus. Furthermore, dTDMA is shown to outperform

---

This work was supported in part by PDG/10 PCI Express, NSF Career 0093085, MARCO/DARPA-GSRC, and SRC 00541 grants.

NoC's in both latency and power consumption in systems with 9 or fewer PE's. While NoC's suffer at high load rates, dTDMA exhibits lower latencies even at full load rate. Such behavior makes dTDMA suitable for high data rate applications, like data streaming and multimedia.

For systems with large numbers of processing elements, bus implementations suffer from scalability issues. Consequently, we propose a new hybrid interconnect that exploits the benefits of dTDMA for small configurations and the scalability of NoC's. In our hybrid interconnect, processing elements exhibiting communication affinity are grouped together on dTDMA buses, while the remaining elements communicate through an NoC. We show that this hybrid dTDMA/NoC system significantly outperforms pure NoC's. Hybrid architectures are already making inroads into current designs. For example, the Cell processor from Sony, Toshiba, and IBM (STI) has 8 on-board synergistic processors using a ring network for communication. A 32-core implementation employing four Cell processors has been proposed using a switched network topology for communication across the processors while still using the ring network within the processor [25]. We anticipate the need for the exploration of similar hybrid interconnect architectures to increase with the proliferation of similar architectures. In this work, we develop a tool geared for this task.

The rest of the paper is organized as follows. In Section 2, we present related work on SoC interconnects. We proceed with an overview of the dTDMA bus architecture in Section 3. In Section 4, we compare our bus architecture with traditional transaction-based buses. We subsequently compare dTDMA to NoC's in Section 5. Section 6 presents the hybrid bus/NoC network architecture and compares it with pure NoC implementations. Finally, we conclude the paper and discuss future research plans in Section 7.

## 2. Related Work

There have been several attempts to improve the performance of on-chip interconnects, including new enhancements to existing standards, like Multi-layer AMBA [3], and research into new interconnect architectures [5][18][20][21] and bus arbitration algorithms [17][20][23].

SAMBA-BUS [20], improves both the architecture and arbitration algorithms of traditional buses. The motivation for SAMBA-BUS is that buses, in their simplest form, waste a significant portion of their wire resources. That is, when two nodes on a bus are communicating with each other, they prevent other nodes from communicating, even if the bus segments required by the communications are disjoint. A common solution to this problem is bus segmentation [8], where a bus is split into segments, each of which can support one communication path. SAMBA-BUS extends this idea, providing a segmentation point at the interface of each node. Furthermore, as an arbitration enhancement, a node may access the bus even if it is not the arbitration winner. However, drawbacks to this bus include a linear twin-bus configuration, which results in a long bus and large capacitive load, and the need for two multiplexers at each node interface.

LOTTERYBUS [17] is an example of an improvement to traditional bus arbitration algorithms. By using a novel "lottery" system (randomized arbitration), LOTTERYBUS is able to reduce latency and provide effective bandwidth

guarantees. This, however, is achieved at the expense of a complicated arbiter.

Though most commonly associated with wireless communication, CDMA has been proposed as a new on-chip interconnection architecture [5][18]. Each transmitter can be active simultaneously by spreading each data bit into multiple "chips" using a unique spreading code. The original data is recovered by the receiver using the same unique spreading code. A major drawback to CDMA is spreading code generation and management. Walsh-Hadamard codes are simple to generate, but their lengths are always a power of two, causing data to frequently be "over-spread," thus wasting bandwidth [19]. Kasami codes are more bandwidth efficient since they can be of any length, but, because they are a decimated version of other codes, generating them on-the-fly is difficult [15]. Additionally, the CDMA channel is multi-valued and must be represented analogously (difficult for VLSI) or digitally (using multiple wires).

Most closely related to our work are [21], [22], and [23]. SiliconBackplane III [23] implements TDMA arbitration using a two-level timing wheel. A fixed size timing wheel contains timeslots for each bus master. If a timeslot is unclaimed by its respective master, SiliconBackplane III will assign other waiting transmitters to the timeslot in round-robin fashion. However, Sonics Inc., the maker of SiliconBackplane III, reports that it can reach up to 90% bandwidth utilization. The dTDMA bus can achieve nearly 100% bandwidth utilization.

Sonics MX [22], the latest interconnect from Sonics Inc., uses a hybrid shared-bus/crossbar approach to improve performance and reduce power consumption. However, it employs a transactional model of communication, which, as we will demonstrate later, introduces inefficiencies.

Niemann et al [21] have proposed a bus/network hybrid specifically for the application of network processors. However, little insight into design decisions and no comparison with existing interconnect technologies are offered.

## 3. TDMA Bus Architecture with Dynamic Timeslot Allocation

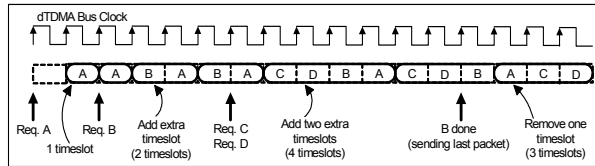
In this section, we outline the operation and implementation of the dynamic Time Division Multiple Access (dTDMA) bus.

### 3.1. Operation of Dynamic Timeslot Allocation

The basic operation of the dTDMA bus is as follows. When a processing element (PE) needs to transmit, it asserts its active signal to the arbiter to request a timeslot. The arbiter then decides (using any number of techniques) on a timeslot assignment for each PE and produces a new configuration for each active transmitter and receiver before the next clock cycle. On the next clock edge, active transmitters and receivers load the new timeslot configuration data and continue operation. When a transmitter is finished communicating, it de-asserts its active signal, and the arbiter de-allocates a timeslot in the same manner that it allocates them. Figure 1 illustrates this operation based on transmission requests.

This method of dynamic allocation always produces the most efficient timeslot allocation – timeslots are never wasted. That is, a timeslot will only exist if it is allocated to a PE, and a

timeslot will never be allocated unless the PE is ready to transmit and requests a timeslot. Because of this, the dTDMA bus is nearly 100% bandwidth efficient (99.9%), thus making the bandwidth utilization equal to the injection rate. The small overhead is due to the initial communication delay of one cycle when a timeslot is allocated.



**Figure 1. Example of Dynamic Timeslot Allocation**

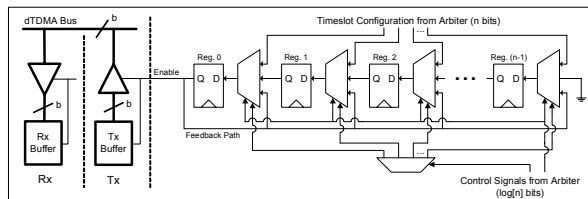
Different methods can be used to assign timeslots to transmitters. The method shown in Figure 1 allocates the first timeslot(s) to the newly-transmitting PE(s). Other methods can schedule timeslots based on the status of transmit buffers or on the length of wait time.

Additionally, various Quality-of-Service (QoS) techniques can be applied to the timeslot assignment algorithm. By assigning a certain percentage of timeslots to a PE, a percentage of bandwidth can be guaranteed. Priority schemes and absolute bandwidth guarantees are also possible by blocking timeslot allocations. Our implementation does not currently include these extensions.

### 3.2. Implementation of the dTDMA Bus

The general architecture of the dTDMA bus is a single broadcast bus connecting all PE's, which have simple transceiver units whose operation is coordinated by an arbiter through a series of control signals. The bus interface (Figure 2) of each PE consists of a transmitter and a receiver connected to the bus through a tri-state driver. The tri-state drivers on each receiver and transmitter are controlled by independently programmed fully-tapped feedback shift registers. Multiplexers between each stage allow for parallel load and a variable length shift register – an  $n$ -stage shift register can be programmed to any length from 1 to  $n$ .

Destinations are represented as a multiple-hot encoding, requiring  $n$  lines (one for each PE). Similarly, the timeslot configuration data requires  $n$  lines. The control signals require  $\log(n)$  lines because they are the select lines for the decoder that sets the feedback path (see Figure 2).



**Figure 2. Architecture of dTDMA Receiver and Transmitter**

During operation, the shift register rotates one stage each clock cycle. The head of the register acts as the enable signal to the tri-state driver. Accordingly, if there are five allocated timeslots and a certain source PE requests bus access and is assigned timeslot 2, the source PE and all destination PE's would receive the initialization data {00100}. Two clock cycles later, the PE's shift registers would contain {10000},

enabling the tri-state drivers, driving the bus, and sampling the bus. On the next rising edge of the clock, the destination PE registers the data before the bus data and shift registers change. The shift registers load new data only when the timeslot allocation changes (i.e. when a PE initiates or ends transmission) and simply operate in rotate mode otherwise.

Because of its broadcast design, the dTDMA bus can easily support multicasting. Multicasting has applications in Multi-Processor SoC's (MPSoC) such as cache coherency protocols and Single Instruction Multiple Data (SIMD) processing. Support for multicasting is simple and inherent in the arbitration protocol, introducing no additional latency.

To support many-to-one communication, first, the transmitting PE's must always prepend data with a start flag which is unique to them. This start flag is always paired with an end flag to signal the end of the transmission. Second, receivers must listen to multiple timeslots; support for this is built into the arbitration protocol. Third, receivers must be able to perform simple message reassembly as they may receive more than one message at a time. This reassembly process is no more involved than what is already required with an NoC interconnect or the AXI bus [2].

Using the Berkeley Predictive Technology Model in a 70nm process [7] and scaled RAW PE tile [26] of square dimension 1.867mm, the maximum frequency of a 9PE dTDMA bus was calculated to be about 916MHz. Buses with 4, 16, and 25 PE's can be clocked at 2.846GHz, 470MHz, and 289MHz, respectively. The reduction of maximum operating frequency with larger buses is due to increasing RC delay.

### 4. Comparison with a Traditional Bus Architecture

There are several different prevalent interconnect structures in MPSoC's, including buses (AMBA, AXI, CoreConnect, STBus) and NoC's (Athereal, STNoC, Crossbow, and Faust) [9].

AMBA [1], AXI [2], and CoreConnect [13] are transaction-based and intended to be used as interconnects between masters (often processors) and slaves (often memories). STBus defines initiators and targets (analogous to masters and slaves) and also uses a transactional model [24]. In this section we will show specific instances in which these transaction-based buses will not perform well, using AMBA as an example.

As a result of its memory-oriented design, AMBA imposes certain restrictions on the nature of addressing behavior. In addition to imposing a 1kB address boundary on sequential transfers, a new transaction must be initiated when the next address to be accessed is not an increment of the previous (non-sequential access). For certain data access patterns, these restrictions may prove to be detrimental to throughput as each new transaction must undergo arbitration and the associated delay. For example, a long transmission (one crossing the 1kB address boundary) will have to be re-arbitrated – at the risk of losing bus ownership – each time it crosses the boundary. Since AMBA arbitrations consume three cycles from request to address transmission and four from request to data transmission, the induced latency overhead may become significant.

In order to operate properly with memories, AMBA transactions occur in two phases – address and data. Once a bus master gains control of the bus, it must first send the

address, and then, in the next clock cycle, send the data. But for processor-to-processor transfers, this two phase transaction is inefficient since it wastes a cycle sending unnecessary address data.

Considering these drawbacks for data streaming and processor-to-processor communication, we chose not to use a memory-oriented or transaction-based bus architecture for the dTDMA bus. Instead, the dTDMA bus is transaction-less and address-mapped. By assigning each PE on the bus a unique identifier (0, 1, 2, etc), PE's can send data to each other without the restrictions of memory-oriented buses. Additionally, multicasting is made trivially simple with an address-mapped scheme.

An example drawback of AMBA as compared to the address-mapped dTDMA bus is that the dTDMA bus only requires a re-arbitration when the destination of a data stream changes. So if PE0 (a processor) is reading data from M1 (a memory), and requests memory addresses that are not sequential or that cross a 1kB boundary, no re-arbitration is necessary since PE0 is still communicating with M1. AMBA would require a re-arbitration for either of these situations. Another advantage of dTDMA is that, in this example, even though PE0 has to re-arbitrate to change destination (e.g. to M2), it will not lose its previously allocated timeslot position, as all the arbiter has to do is reprogram the receivers (M1 and M2). Hence, there is no effective arbitration delay even though a change of destination has occurred.

A simple, illustrative, comparison between AMBA and dTDMA appears in Figure 3. In this scenario, master A requests access to the bus on clock T1, followed by master B requesting access on clock T2. Master A does not receive a grant from the AMBA arbiter until clock T3, while the dTDMA arbiter issues a new timeslot configuration before the end of clock T1. Data transmission commences with clock T5 in the case of AMBA, since T4 is dedicated to the address phase of the transaction. dTDMA allows data transmission to commence earlier, at T2. Master B waits five clock cycles from request to data transfer in the AMBA bus, yet only one cycle in the dTDMA bus. In this scenario, the dTDMA bus completes the transmission of two words from each master three cycles before the AMBA bus does.

Even though dTDMA does not explicitly support memory transfers, implementation is not difficult. For a read operation, the address is sent to the memory element during the first occurrence of the timeslot. The memory element latches the address, and has the data available by the time the timeslot rolls around again. Write operations can be performed by piggybacking the address with the data, so that, when the timeslot rolls around again, the memory has completed the write and is ready to accept a new transfer.

dTDMA has the attractive quality of predictable latencies because a PE is guaranteed to wait no longer than the number of active transmitters. In contrast, the AMBA bus may make masters wait an undetermined amount of time before they are granted ownership of the bus. Because of its predictable latencies, the dTDMA bus is also non-blocking, that is, a PE will never be *denied* access to the bus on account of another PE monopolizing resources. Both of these qualities are important when designing to meet the real-time constraints of multimedia processing applications.

By allocating bus resources as streams instead of transactions, arbitration overhead is reduced for long transmissions. Moreover, the dTDMA arbitration-to-

transmission delay is only one cycle, compared to four (for data) in AMBA. The overall effect of moving to a transaction-less, address-mapped bus for streaming data is that fewer arbitrations take place and each can be done more quickly, and a cycle is not wasted in sending address data when it is not needed.

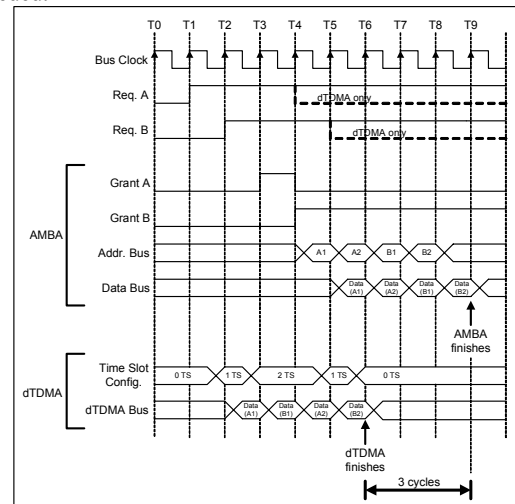


Figure 3. A Simple 2-PE Transfer on dTDMA and AMBA

## 5. Comparison with Networks-on-Chip

### 5.1. Interconnect Size

One criterion for choosing a bus or an NoC is the number of PE's present in the system. Capacitive loading and contention – problems that limit the performance of large buses – are mitigated by an NoC interconnect. In this section, we compare the scalability of the dTDMA bus and NoC's.

### 5.2. Experimental Setup

A state-of-the-art NoC design appearing in [14] uses a link width of 128. A bus width of 512 bits was chosen for the dTDMA bus as the equivalent comparison for the following two reasons.

First, each NoC router has the ability to send *and* receive data on each of its four links (North, South, East, and West) at once. This means that with each clock cycle, a 128-bit NoC router can transmit 512 bits to its neighboring routers, and receive 512 bits from neighboring routers. Since the bus is inherently half-duplex, to mimic the NoC abilities we would need two 512-bit buses. However, this is excessive as the bus would be severely underutilized in low traffic conditions. It will be shown later that, even with a 256-bit bus, dTDMA significantly outperforms an NoC with 128-bit links.

Second, the dTDMA bus width can be increased without suffering decreased speed or increased complexity. On the contrary, increasing the bit width in an NoC greatly affects three crucial architectural components within the node router: the input buffers, the virtual channel logic and buffers, and the switch crossbar. Specifically, an NoC can not scale easily for the following reasons: 1) architectural components of each router will increase in size with bit width, 2) router components, in addition to increasing in size, will consume

more power as bit width increases, and 3) each router is replicated  $n$  times in an  $n$ -PE mesh NoC. Using a Verilog implementation of an on-chip router implemented in 90nm TSMC libraries, we found that doubling the bit width of the NoC incurred an area increase of 55% and a power increase of 77% per router. Additionally, since each router has five connections, each  $2b$  bits wide ( $b$  bits in each direction), doubling the bit width incurred a ten-fold increase in wiring complexity. The router in our work was modeled based on the low latency NoC router of [14].

To model the dTDMA bus, a cycle-accurate simulator was written in C++. The simulator is fully configurable; using a configuration file to describe the operational parameters of each PE. Specifiable parameters include data injection mode (file input, uniform, multimedia, master, slave), injection rate, destination mode (file input, uniform, static, multicast, master, slave), and static destination lists.

A modified version of the simulator from [14] was used in conjunction with the dTDMA simulator to produce the results in the following section. In the NoC simulator, routers have twelve virtual channels each with a four-flit (one message) buffer, and an input buffer of 32 flits (8 messages) from the local PE connection. Accordingly, the dTDMA simulator uses an 8-message transmit buffer on each PE. A frequency of 500MHz was used in both simulators to estimate power consumption. Injection rate is defined as the number of 512-bit messages injected per PE per clock cycle. Both simulators used a uniform random destination pattern.

Two types of traffic injection were used – uniform and multimedia injection. In uniform injection, data arrives with a constant inter-arrival time, implying constant data rate. Multimedia traffic injects data with varying inter-arrival times based on real MPEG2 traces.

### 5.3. Experimental Results

Using the setup described above, experiments were carried out to find the average latency and power consumption of different size dTDMA buses and NoC meshes with respect to load rate.

With uniform injection (Figure 5(a-d) on page 8), the dTDMA bus consistently outperforms the NoC mesh for PE populations of 4, 9, 16, and 25. NoC performance degrades more severely with injection rate than the dTDMA bus. However, at higher numbers of PE's, the dTDMA latency nears that of the NoC. These results indicate that dTDMA mitigates the contention issues, which afflict traditional bus architectures. Hence, dTDMA allows for up to 25 PE's to be placed on a bus without excessive contention problems. However, capacitive loading remains a limiting factor. It was shown in Section 3.2 that the maximum bus clock frequency drops below 500 MHz beyond 9 PE's. Thus, for designs operating at or below 500 MHz, dTDMA will still outperform traditional bus architectures and NoC's even for PE populations of more than 9. To take advantage of both low latencies and high clock frequencies, though, we conservatively conclude that the dTDMA bus performs optimally with PE populations of 9 and fewer. This result is in agreement with other evaluations of practical bus size [6]. Even at 9PE's, the dTDMA bus shows a 66% reduction in latency over the NoC.

Multimedia traffic results, shown in Figure 5 (e-h) on page 8, exhibit trends similar to those of uniform traffic. A

slight decrease in latency with increasing injection rate in (e) is visible because, at lower load rates, PE's do not transmit often enough to retain their timeslots and frequently must re-arbitrate to get a new timeslot. The same phenomenon, compounded with the large bus size, causes the 25 PE bus to perform slightly worse than the NoC, as shown in Figure 5 (h).

Results for a 256-bit bus are similar for 9 and fewer PE's. Though worse in absolute latency compared to the 512-bit bus by 32%, the results were still better than the NoC for all load rates. Thus, given equal *wiring area* (each side of the NoC router has 2 128-bit links), and discounting the router area overhead of the NoC, the dTDMA bus still outperforms the NoC.

Similarly, equal wiring area can be achieved by doubling the NoC bit width instead of halving the dTDMA bus width. Hence, each side of the NoC router would have 2 256-bit links. This scenario was also simulated with the NoC showing a 25% improvement in performance. This NoC improvement was not enough to outperform the 512-bit dTDMA bus at any load rate.

Power consumption was evaluated by implementing the dTDMA architecture in Verilog and synthesizing using 90nm TSMC libraries. Power values for individual components, as reported by Synopsys Design Compiler, were imported into the dTDMA simulator. Based on actual component utilization, the total power consumption was calculated. The bus consumes less power for all injection rates and traffic types, most likely due to the lack of large packet buffers and crossbar switches, which are present in each NoC router.

## 6. dTDMA / NoC Hybridization

We now have two heuristics to use when deciding how to connect the PE's in an SoC: 1) Buses with 9 or fewer PE's perform better than NoC's of equal size; and 2) NoC performance degrades much faster than bus performance with increasing load rate.

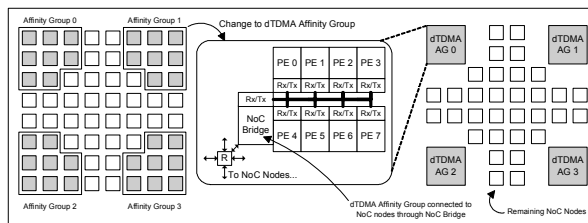
Traffic hotspots, caused by localized congestion, in NoC's have been observed in previous studies [10]. These hot spots will increase the overall latency of communication in the entire network, so they should be avoided when possible. The process of affinity grouping is one way to accomplish this. Affinity grouping attempts to cluster frequently-communicating PE's together in order to reduce overall hop count. Such grouping has been effectively utilized in other SoC applications, such as wireless decoding [12] and video applications [4]. We refer to this type of NoC grouping as an AG Mesh.

However, simply placing PE's in an affinity group will not decrease the traffic load on the network locally, within the affinity group, so the network will still exhibit high latencies due to link contention despite a lower hop count. But we now have a solution for this problem: the dTDMA bus. Heuristics (1) and (2) lead to the conclusion that the dTDMA bus will perform much better as a local affinity group interconnect than an NoC, as the dTDMA bus lends itself better to small clusters with high data rates (see results from Figure 5). Communication with PE's outside of the affinity group is accomplished with the assistance of a local bridge that packetizes data for transmission on the NoC backbone. The process of hybridizing the AG Mesh, as shown in Figure 4,

produces the architecture that we call the dTDMA/NoC hybrid.

Figure 4 shows both the AG Mesh architecture (left) and the result of hybridization with the dTDMA bus (right). The shaded squares clustered in the corners of the AG Mesh represent the PE's of the affinity groups. All PE's (including those in the affinity groups) are connected by a single NoC in the AG Mesh, but after hybridization, affinity groups are connected with a dTDMA bus and each affinity group is connected to other affinity groups and non-affinity group PE's through an NoC bridge.

Multicasting is easily supportable inside an affinity group after hybridization. NoC's are currently unable to support multicasting without greatly increasing network traffic or making each router significantly more complex.



**Figure 4. Hybridization of an AG Mesh**

To illustrate affinity grouping, we present an example MPSoC – a modern cable or satellite Set-Top Box (STB). Such a multimedia system would simultaneously decode the video feed (digital or analog), the audio feed (mono, stereo, multi-channel) and data for content-rich video streams (interactive programming, electronic program guides, etc.). A possible architecture for this system could be as follows: 1) A central demultiplexer that pulls out the audio, video, and data streams sits outside of any affinity groups, since it is shared by many PE's; 2) Decoding of the audio, video, and data streams is done by specialized systems inside separate affinity groups; 3) Decoded video and data streams are combined to form the final picture in a D/A converter sitting outside of affinity groups; and 4) A central controller core performs synchronization, low-bandwidth I/O, and system initialization. It also sits outside of the affinity groups. A distributed system such as this will exhibit varying degrees of affinity depending on the incoming data. For example, a variable bit rate video decoder will receive different data rates depending on the compressibility of the video stream, but will always output the same bit rate.

## 6.1. Experimental Setup

We modified both the dTDMA and NoC simulators and integrated them, defining a new parameter called the “bridge rate,” which is the percentage of data generated by each PE in an affinity group that is destined for a PE outside of the affinity group. We varied this parameter from 10% to 50% to simulate the dynamic nature of traffic in multimedia SoC's. The dTDMA simulator allows one PE to be designated as a bridge and allows the rate at which other PE's send and receive data to and from the bridge to be specified.

While a bridge rate of 50% may seem extreme (and in fact warrants removing a PE from an affinity group), due to the dynamic nature of some processing systems – e.g. in advanced multimedia embedded systems like the STB described in the introduction of this section – the bridge rate

may occasionally peak near 50%, and so it is important to model this behavior.

Our test bed was a large 8x8 PE SoC with four affinity groups. Affinity groups of 8PE's plus one bridge PE were chosen for the hybrid because of the 9-PE dTDMA bus size established in Section 5.3. To fairly model the same type of grouping in the AG Mesh, affinity groups of 8PE's were used and the injection rates of four PE's were set to zero to account for the bridges in the hybrid system (which do not generate data themselves, but merely relay it).

## 6.2. Experimental Results

Results for both uniform and multimedia traffic are shown in Figure 6 (a-f) on page 8. In all test cases, the dTDMA/NoC hybrid outperformed the AG Mesh by a sizeable margin. The worst-case latency reduction is 15.2% with multimedia injection and a bridge rate of 10%. Interestingly, the latency gap between the AG Mesh and the hybrid increases as the bridge rate rises; probably the result of increasing contention on NoC links near the periphery of the affinity groups.

Across all bridge rates, the latency behaviors of the AG Mesh and hybrid are similar to the standalone NoC and dTDMA systems. However, saturation is reached much more quickly than in the standalone systems because latency is now dominated by the NoC backbone, which saturates faster than the dTDMA buses.

In the AG Mesh, the absolute latency increases with bridge rate because more data is sent to destinations outside of the affinity groups and the NoC routers on the edge of the affinity groups become congested with data destined for foreign PE's. Similarly, in the hybrid architecture, as bridge rate increases, the bridges become congested, and the absolute latency increases.

A solution for mitigating bridge contention in the hybrid system is to use more than one bridge per affinity group. To avoid complicated bridge arbitration in this scenario, PE's can be statically assigned a specific bridge to use.

With regards to power consumption, the dTDMA/NoC hybrid resulted in an 8% to 27% decrease over the AG Mesh. Results are omitted due to space considerations.

## 7. Conclusions & Future Work

In this paper, we have demonstrated a need for transaction-less buses for high data rate on-chip applications. A new bus architecture, dTDMA, was proposed, and through analysis, it was shown to be more efficient than AMBA, especially for long transmissions.

We also showed that dTDMA exhibits at least 66% lower latencies and a 20% average reduction in power than a state-of-the-art NoC architecture with fewer than ten PE's. To combine the positive attributes of smaller buses and the scalability of NoC's, we proposed a hybrid dTDMA/NoC affinity grouping architecture intended for large SoC's. Latency can be reduced by grouping frequently-communicating PE's into an affinity group and placing them adjacent to each other. Using a dTDMA bus as the local interconnect inside the affinity groups results in a latency reduction of at least 15.2% and an average power reduction of 12% over a pure NoC.

The simulator developed for this work helps fill a void in the analysis of hybrid interconnects. Currently, the simulator integrates the dTDMA bus with NoC's. An extended version of the simulator supporting additional interconnect architectures, such as rings, switches, and crossbars, is being investigated. Other areas for further research include a quantitative analysis of multicasting and its effects on affinity grouping, the effect of different dTDMA arbitration policies on latency, and QoS techniques for the dTDMA bus architecture. Additionally, there are a number of traditional bus enhancements like low voltage-swing drivers and segmentation that can be applied and studied in the context of the dTDMA bus architecture.

## 8. References

- [1] ARM, "AMBA Specification Rev. 2.0," [www.arm.com](http://www.arm.com), 1999.
- [2] ARM, "AMBA AXI Protocol v 1.0 Specification," [www.arm.com](http://www.arm.com), 2004.
- [3] ARM, "Multi-layer AHB, Overview," [www.arm.com](http://www.arm.com), 2001.
- [4] Artieri, "NOMADIK: An MPSoC Solution for Advanced Multimedia," *Proceedings of the 5th International Forum on Application Specific MPSoC*, July 2005.
- [5] Bell, et al, "CDMA as a Multiprocessor Interconnect Strategy," *Asilomar Conference on Signals, Systems and Computers*, vol 2, pgs 1246-1250, 2001.
- [6] Benini and De Micheli, "Networks on Chips: A New SoC Paradigm," *IEEE Computer*, pgs 70-78, 2002.
- [7] <http://www-device.eecs.berkeley.edu/~ptm/interconnect.html>
- [8] Chen, et al, "Segmented Bus Design for Low-Power Systems," *IEEE Transactions on VLSI Systems*, vol 7, no 1, pgs 25-29, 1999.
- [9] Coppola, "From Spaghetti wires to NoC," *Proceedings of the 5th International Forum on Application Specific MPSoC*, July 2005.
- [10] Dally and Towles, *Principles and Practices of Interconnection Networks*, Morgan Kaufmann, 2004.
- [11] Dally and Towles, "Route Packets, Not Wires: On-Chip Interconnection Networks," *Proceedings of ACM DAC*, pgs 684-689, 2001.
- [12] Hung, et al, "Thermal-Aware IP Virtualization and Placement for Networks-on-Chip Architecture," *Proceedings of the International Conference on Computer Design*, October 2004.
- [13] IBM, "32-bit Processor Local Bus. Architecture Specifications. Version 2.9," [www.ibm.com](http://www.ibm.com), 2001.
- [14] Kim, et al, "A Low Latency Router Supporting Adaptivity for On-Chip Interconnects," *proceedings of ACM DAC*, 2005.
- [15] Komo and Liu, "Modified Kasami Sequences for CDMA," *Southeastern Symposium on System Theory*, pgs 219-222, 1990.
- [16] Krewell, "Multicore Showdown," *Microprocessor Report*, pgs 41-45, vol 19, May 2005.
- [17] Lahiri, et al, "LOTTERYBUS: A New High-Performance Communication Architecture for System-on-Chip Designs," *ACM DAC*, pgs 15-20, 2001.
- [18] Lai, et al, "CT-Bus: A Heterogeneous CDMA/TDMA Bus for Future SOC," *Asilomar Conference on Signals, Systems and Computers*, vol 2, pgs 1868-1872, 2004.
- [19] Lee and Miller, *CDMA Systems Engineering Handbook*, pgs 425-460, Artech House Publishers, 1998.
- [20] Lu and Koh, "SAMBA-BUS: A HIGH PERFORMANCE BUS ARCHITECTURE FOR SYSTEM-ON-CHIPS," *Proceedings of ACM ICCAD*, pgs 8-12, 2003.
- [21] Niemann, et al, "A Scalable Parallel SoC Architecture for Network Processors," *IEEE Annual Symposium on VLSI*, 2005.
- [22] Sonics Inc., "Sonics MX," <http://www.sonicsinc.com/sonics/products/smx/>
- [23] Sonics Inc., "SiliconBackplane III," <http://www.sonicsinc.com/sonics/products/siliconbackplaneIII/>
- [24] STMicroelectronics, "STBus Communication System," [http://mcu.st.com/files/mcu/STBus\\_spec.htm](http://mcu.st.com/files/mcu/STBus_spec.htm), 2003.
- [25] Su, "Digital Media: The New Frontier for Supercomputing," *Proceedings of the MPSoC Conference*, July 2005.
- [26] Taylor, et al, "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs," *IEEE Micro*, pgs 25-35, March-April 2002.

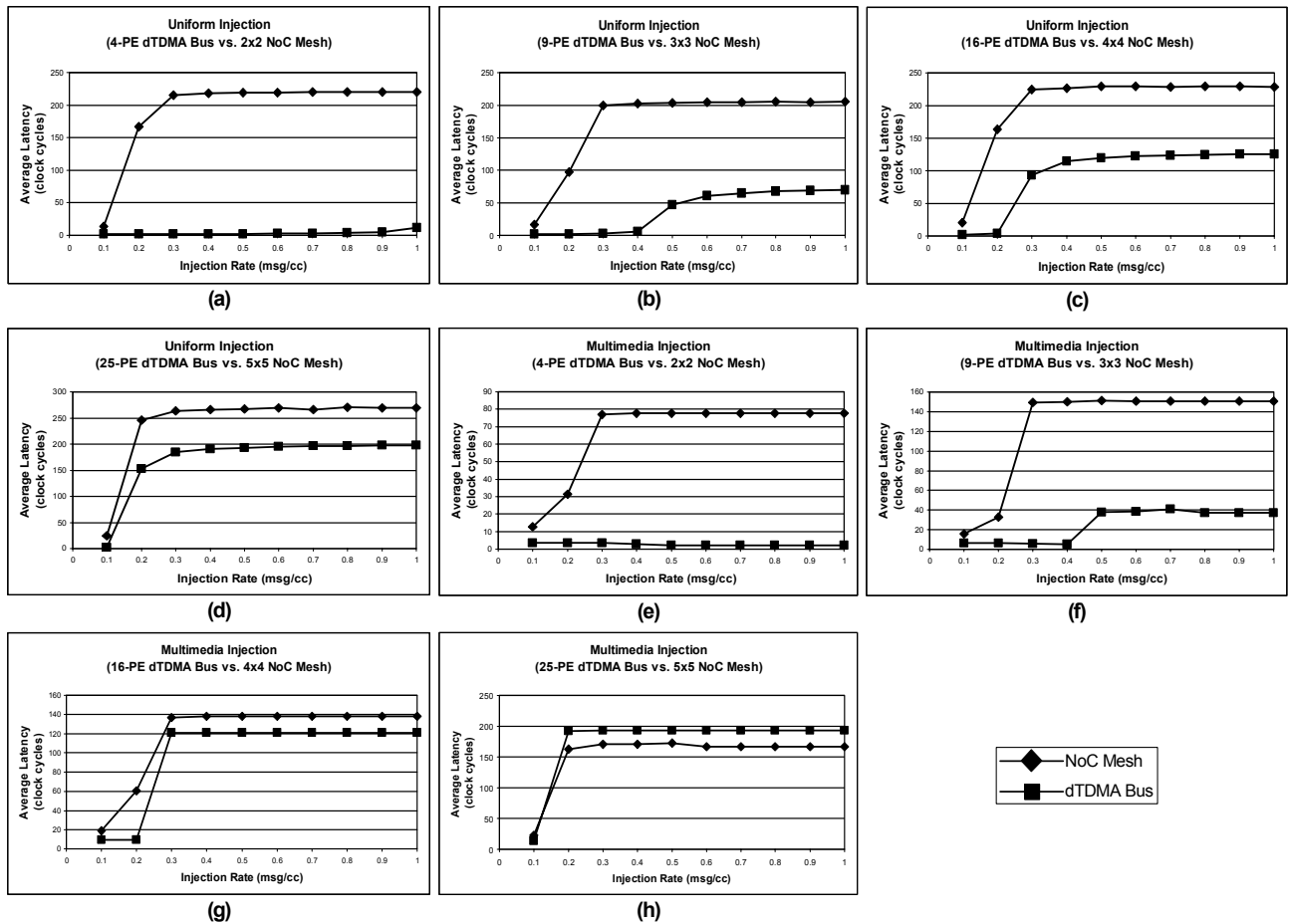


Figure 5. Standalone dTDMA and NoC Latencies for Uniform (a-d) and Multimedia Traffic (e-h)

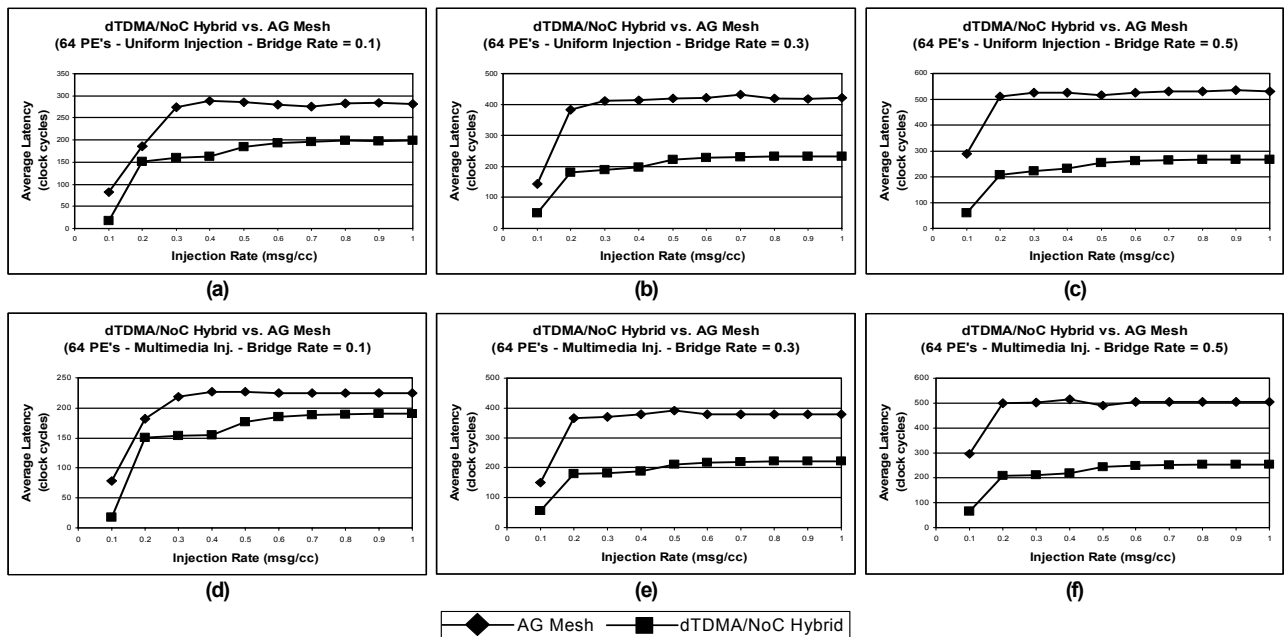


Figure 6. Hybrid dTDMA/NoC and AG Mesh Latencies for Uniform (a-c) and Multimedia Traffic (d-f)