SoCBUS: Switched Network on Chip for Hard Real Time Embedded Systems

Daniel Wiklund and Dake Liu
Dept. of Electrical Engineering
Linköping University
S-581 83 Linköping, Sweden
{danwi,dake}@isy.liu.se

Abstract

With the current trend in integration of more complex systems on chip there is a need for better communication infrastructure on chip that will increase the available bandwidth and simplify the interface verification. We have previously proposed a circuit switched two-dimensional mesh network known as SoCBUS that increases performance and lowers the cost of verification. In this paper, the SoCBUS is explained together with the working principles of the transaction handling. We also introduce the concept of packet connected circuit, PCC, where a packet is switched through the network locking the circuit as it goes. PCC is deadlock free and does not impose any unnecessary restrictions on the system while being simple and efficient in implementation. SoCBUS uses this PCC scheme to set up routes through the network. We introduce a possible application, a telephone to voice-over-IP gateway, and use this to show that the SoCBUS have very good properties in bandwidth, latency, and complexity when used in a hard real time system with scheduling of the traffic. The simulations analysis of the SoCBUS in the application show that a certain SoCBUS setup can handle 48000 channels of voice data including buffer swapping in a single chip. We also show that the SoCBUS is not suitable for general purpose computing platforms that exhibit random traffic patterns but that the SoCBUS show acceptable performance when the traffic is mainly local.

1. Introduction

System-on-chip (SoC) designs pose many challenges on the designers. Design of a complex SoC with limited time and resources is a major hassle in the industry already today. With further technology progress and the ever increasing demands on integration there are a number of problems that quickly become big issues, e.g. design time and verification.

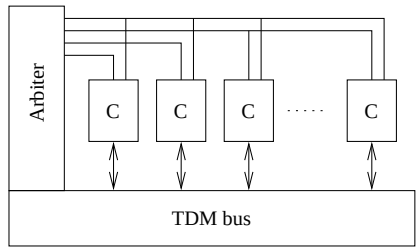
The major direction out of some of these problems is to make use of (both external or internal) intellectual property (IP) blocks, i.e. complete subsystems - like a microcontroller or a DSP - that can be integrated on chip together with other IP and custom logic to make up the complete system-on-chip product.

As the number of IPs on a chip increases there is an increasing demand of high performance communication between the IPs. The current solution when implementing SoCs with multiple processors, ASICs, memories, etc. is to use a time-division multiplex (TDM) bus, e.g. the AMBA bus from ARM, Inc [1]. The fundamental limits in this kind of bus is the arbiter used for a multi-master setup and the shared media.

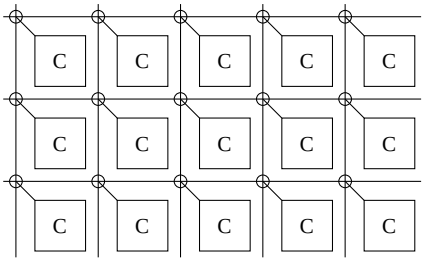
Another problem when designing complex chips using IP blocks is the verification of the interfaces between the IPs, the communication structure, and the custom "glue" logic generally used to connect them. To alleviate the problems of verification and performance we have proposed a two dimensional mesh network, called the Linköping SoCBUS [2].

This paper is divided into three main parts. Sections 2 through 4 discusses general issues concerning SoC networks as a concept as well as the proposed SoCBUS with network architecture and transaction handling. Section 5 introduces an example of a possible hard real-time application, the PSTN/VoIP gateway. Sections 6 and 7 briefly introduces the simulator and discusses the stimuli and simulation results for our SoC network using random pattern traffic and the PSTN/VoIP gateway example. Finally the paper is closed with a description of some future work and the conclusions in sections 8 and 9.

This work is supported by the Integrated Electronics (INTELECT) program of the Swedish Foundation for Strategic Research (SSF). Daniel Wiklund is a Ph.D. student at the division of Computer Engineering and Dake Liu is the chairman professor of the division of Computer Engineering. Authors would like to thank Erik Svensson for his work in this project.



(a) Traditional bus



(b) Network bus

Figure 1. Comparison of traditional TDM bus and a switched interconnect bus replacement

2. The SoCBUS as a bus replacement on chip network

Since the shared TDM bus is becoming a bottleneck many researchers and companies have realized the need of a better way of handling the on-chip communications. With the widespread knowledge in general purpose computer networks at hand, researchers saw a great possibility of using networks not only for off-chip communication but also for on-chip communication. fig. 1(a) shows a traditional TDM bus setup and fig. 1(b) show the switched network counterpart. It is obvious even from a quick glance that the possible performance is significantly higher for the network based solution.

Several research groups around the world are currently involved in research on on-chip networks. A common feature among almost all of these research projects is that they use packet switching [3], [4] following the seven layer OSI model [5].

The seven layer OSI model was developed with general purpose networks in mind and is not necessarily applicable directly to all types of networks. One such network type where the layered approach of implementation is unsuitable is on-chip networks. The reason is that there is some fundamental differences between a general purpose network (e.g. Internet) and an on-chip network. The general purpose network is in principle unknown, i.e. nodes can be added and removed at any time, the topology is not fixed, and so on, while the on-chip network is well-known since it is fixed at the time of chip manufacturing.

Table I. The seven OSI layers and their coverage in the SoCBUS system

7	Application layer	Not covered by SoCBUS
6	Presentation layer	Can be covered by wrappers
5	Session layer	Can be covered by wrappers
4	Transport layer	Covered by SoCBUS
3	Network layer	Covered by SoCBUS
2	Link layer	Covered by SoCBUS
1	Physical layer	Covered by SoCBUS

With this additional knowledge of the network infrastructure it is possible to group several layers of the OSI model and therefore simplify the hardware and software while at the same time cut latency in the network. Our proposed solution has a broad coverage of the OSI-model, see table I, with a minimum of complexity.

A network using circuit switching has low complexity switching nodes because their main function only is to connect an incoming link to an outgoing link. Deadlock avoidance is easily achieved since the circuit setup can either succeed or fail but it cannot stall somewhere in the process. Packet switching inarguably leads to more complex switching nodes since the node in order to avoid deadlocks either have to buffer every packet in its entirety before routing it to the next node, the node have to use several virtual channels [6], or the node has to restrict the possible paths [7].

Packet switching also suffers from latency problems where the packet delay through the network can be several hundred or even several thousands of cycles dependent on routing algorithms and switch implementations [8]. One example is where a fully buffering implementation of a packet switched network transfers 128 bytes of information between two diagonally opposite corners of a 8x8 mesh. The time spent in the network would then be more than $128 \cdot 15 = 1920$ cycles. Even for a wormhole routing network there is a possibility that the packets will be stalled for long times due to other traffic. This is because there is always a statistical distribution of packet delays in a packet switched network which also creates the possibility for packets to arrive out-of-order to the desti-

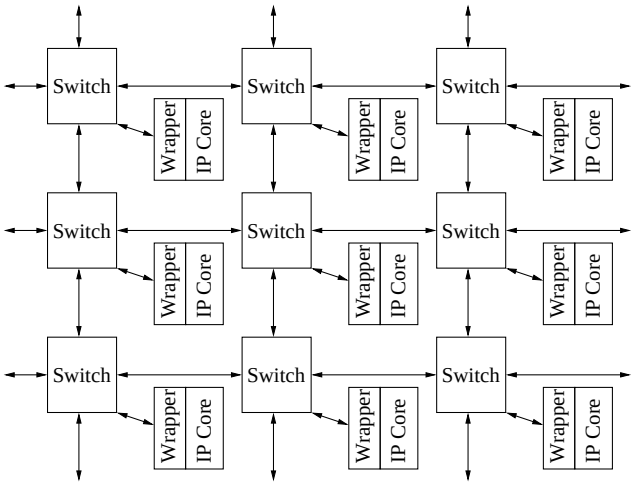


Figure 2. A 4x3 node switched network with wrappers and IP blocks

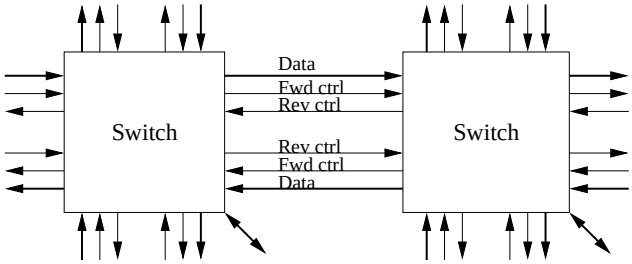


Figure 3. Interface between switching nodes and wrappers within the network

nation. Circuit switching has a clear advantage over packet switching since the data latency is only dependent on the distance and there is no dependency on other factors, e.g. other traffic in the network. All data is also guaranteed to arrive in the same order as sent. The only dependency on the traffic situation in a circuit switched network is when setting up a route.

3. SoCBUS architecture

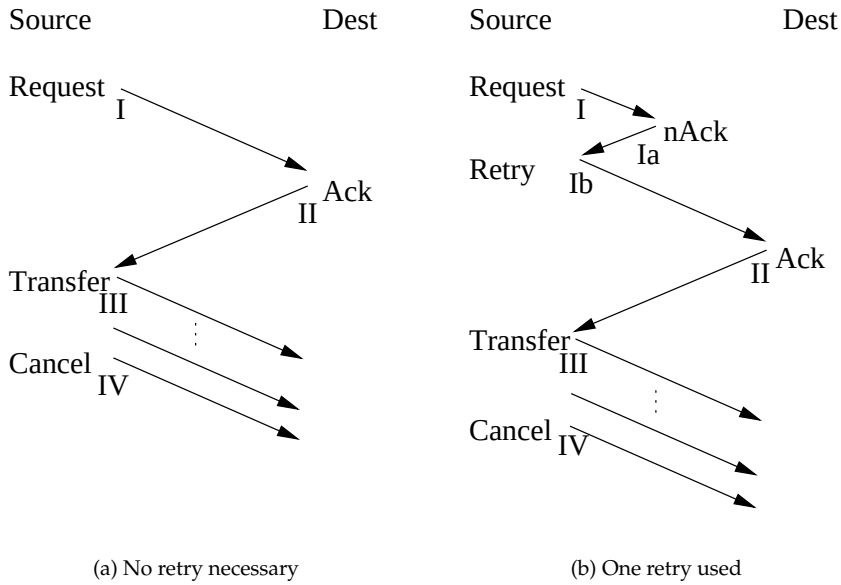
A thorough investigation of different network topologies were conducted [9] and a two dimensional mesh was considered to be the most suitable for on-chip networks. This is also the common topology proposed by most other researchers [3], [4], [10]. The main reasons for selecting a the two dimensional mesh instead of other topologies such as hexagons, butterflies, or tree are that a two dimensional mesh have an acceptable wire cost, reasonably high bandwidth, and that it is easy to group IPs that communicate a lot so that they do not consume an unnecessary high amount of resources in the network.

The network, see fig. 2, uses five port switches that allocate four ports to connect to adjacent switches and one port to connect to the local IP block interface (port). The local IP port is connected through a wrapper to the local IP block. The wrappers handle IP and network port differences such as transaction handling, port width, endianness, etc. The wrappers also contain any necessary buffers and does the bridging between the IP block clock domain and the network clock domain.

The interfaces internal to SoCBUS all use the same physical format, see fig. 3. In total 11 wires are used in each direction where eight wires carry forward going data and routing request packets, one wire is used for forward control, and two wires are used for reverse control. The forward control handles framing of transmissions and clock information to allow for easy retiming of the transfers when using mesochronous clocking (i.e. same frequency but unknown phase) [9]. The reverse control carry the acknowledgments. The diagonal lines in the figure represents an identical interface to the local IP wrapper port.

The simplicity of the SoCBUS components allow for an implementation of switches and the network side of wrappers with very low logic depth, i.e. 6-8 gate delays. With this low logic depth it is possible to run the SoCBUS at 1.2 GHz in a 0.18 micron technology process. This is four times the maximum expected IP block clock frequency of 300 MHz in the same process. This difference in clock rates will further mask the latency since a four cycle latency in the network will look like an one cycle latency to a core.

Considering the high clock rate and the distributed nature of an on-chip network, the wire delays between the components become a serious problem if using traditional synchronous design methodology. In order to allow for wire delay and skew we propose the use of mesochronous (i.e. same frequency but unknown phase) clocking with signal retiming in the system. Using mesochronous clocking and retiming still requires the wires delays within a link to have reasonable skew but allows the design to use links that have differing delays without any problem. To further simplify the connection of network components we propose using optimized drivers and transmission-style wires [11], [**?**]. This gives many advantages, e.g. no repeaters are necessary, the system consumes a minimum of power, and there is no need of laying out the network in an ordered fashion on chip.



(a) No retry necessary

(b) One retry used

Figure 4. Two successful circuit setups

4. Network transaction handling

4.1. Route setup flow

The network transactions consist of four to six phases dependent on whether the first try routing is successful or not. A successful transaction, see fig. 4(a), has four phases. (I) First a request is sent from the source to the network. As this request finds its way through the network the route is temporarily locked and cannot be used by any other transactions. (II) The second phase starts when the request reaches its destination an acknowledge is sent back along the route and the locks are changed to permanent locks (referring to the current transaction time span only). (III) When the acknowledge has returned to the source the third phase starts. This phase holds the actual transfer of data payload. (IV) Finally after the data has been transferred a cancel request is sent that releases all resources as it follows the route.

If a route is blocked in a node the routing request is canceled by (Ia) the blocking switch returning a negative acknowledge to the source, see fig. 4(b). (Ib) The source must then retry the route at a later stage which means that the additional two phases (nAck and retry) might need to iterate.

4.2. PCC: Packet connected circuit

We refer to the novel hybrid circuit switching with packet based setup introduced in the previous subsection as "packet connected circuit" or PCC for short. The PCC has very nice properties in several areas as follows:

- The PCC is deadlock free since no resources are locked while waiting (indefinitely) for other resources.
- The routing hardware in the switching nodes become very simple when no special cases, stalls, or virtual channels must be considered.
- Need for a minimum of buffering capable of holding just a request package is needed in the switching nodes.
- PCC gives the lowest possible latency of just one retiming latch pipeline at each switching node.
- There is no inherent limit on route selection algorithms in the PCC scheme.

4.3. Routing

After studying publications on routing algorithms and performing some simple experiments it was decided to go for a simple minimum path length routing. Each switch has the knowledge of the general direction to each destination, i.e. north, west, south, east, down, and combinations of these, e.g. north-west. Since the network do not change when the chip has been designed this knowledge is static and decided at the time of network layout design. The routing decisions are simply based on the destination address and the known direction. If there are more than one direction that leads to the destination one is selected ac-

cording to a round robin scheduling. If the primary selection is occupied the second choice will be used instead. If there are no free outputs that lead towards the destination the routing will fail and the switch will return a negative acknowledgment to the source.

5. Application: The PSTN / VoIP gateway

As an example of a system where an on-chip switched network is suitable we have selected a gateway connecting between both the public switched telephone network (PSTN) and voice-over-IP (VoIP). The data flow architecture for such a gateway chip is shown in fig 5. The architecture supports PSTN to PSTN, PSTN to VoIP, VoIP to PSTN, and VoIP to VoIP connections. In addition to this it also supports echo canceling and A-law/u-law companding. The connections to the chip is typically T1/E1 (24/30 channels) or even up to OC-3 (capable of carrying 2048 channels) for PSTN and Ethernet for VoIP. This hardware architecture is used as a basis for the software architecture, see fig 6, and has been suggested by the industry [13].

We propose using the SoCBUS switched interconnect for dynamic connections of program loading, payload delivery, computing buffer passing, and control signaling in this architecture. A simplified version of the architecture has been used for simulations, see section 7.2, to show the suitability of SoCBUS as an interconnect structure for hard real-time systems.

6. Simulation environment

In order to assess the performance of our proposed network we have developed a cycle true behavioral simulator for our SoC network and showed the usefulness both in the research as well as the customer design flow [9]. This behavioral simulator is event driven and implemented in C++.

The simulator uses models of the state machine in the switching nodes and the network end of the wrappers. A slightly simplified state chart for the specific implementation of the PCC scheme state machine for a switching node used in our simulations can be found in fig. 7. The switch starts up in the idle state (1) waiting for a request. When a route request packet is received the switch tries to find a route (2) that is available and leads towards the destination. If such a route is not found a negative acknowledgment is returned to the sender (3). If a route is found this is locked (4) and the switch passes the request packet on via the output (5). The switch then waits for an acknowledgment to be received from the output (6). If a negative acknowledgment is received the switch will release the lock and pass on the negative acknowledgment (3). If the result were a positive acknowledgment the switch passes the acknowledgment on to the source (7) and enters the transfer state (8). When the transfer is finished and the request cancel is received from the source the switch will release the lock (9) and return to idle. As can be seen in the figure, the switch FSM states are closely related to the transfer transaction phases.

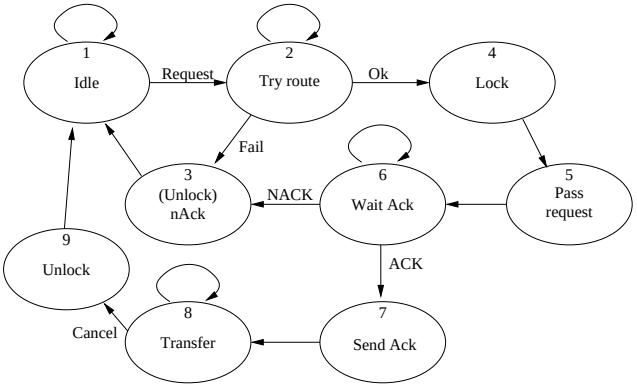


Figure 7. State diagram for the FSM implemented in the switching node

6.1. Latency analysis

There are two primary types of latency in the network. One is related to the route setup time and one is related to the payload transfer. Both latencies are linearly dependent on the distance between source and destination. The route setup latency consists of first the request handling latency that is minimum of four bus cycles per switch for request buffering and route selection. The second part of the route setup latency is the acknowledgment latency that is one cycle per switch.

Since the network is circuit switched the data transfer latency is just one cycle per switch to allow for retiming.

Due to the high internal clock rate in the SoCBUS the latency will appear lower from the IP block perspective. With a SoCBUS clock of 1.2 GHz and the typical IP block clock frequency of 300 MHz the apparent latency will be only a fourth. The route setup latency for every switch will appear as 1 cycle and the data transfer latency as 0.25 cycles.

7. Stimuli and simulations

7.1. Random pattern stimuli

Two sets of stimuli has been used in the simulations of the network. The first stimuli is random pattern communication with different mean usage levels.

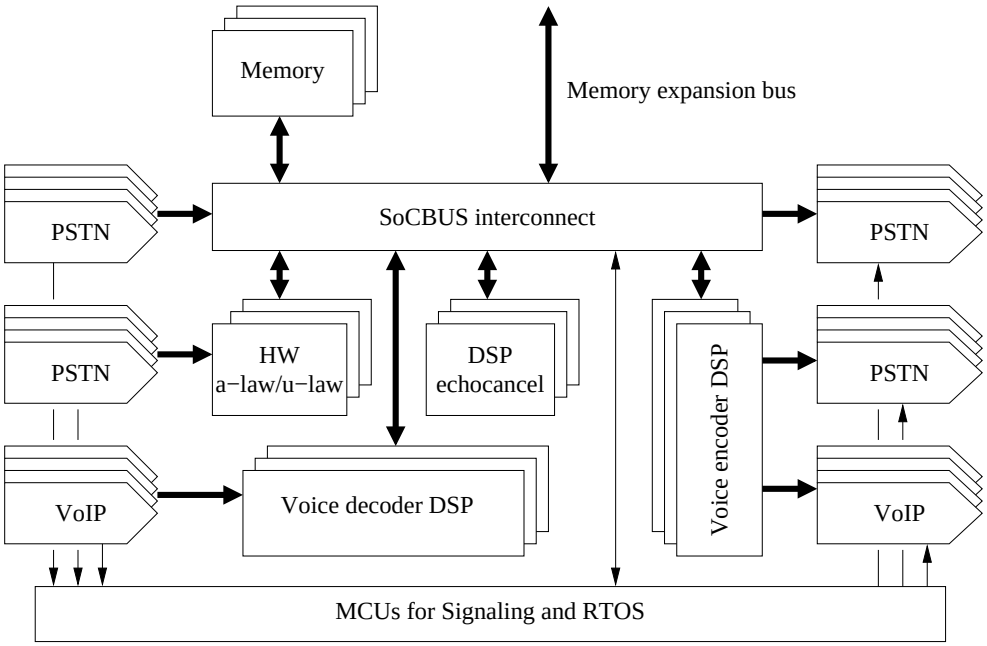


Figure 5. Data flow architecture of a PSTN/VoIP gateway chip

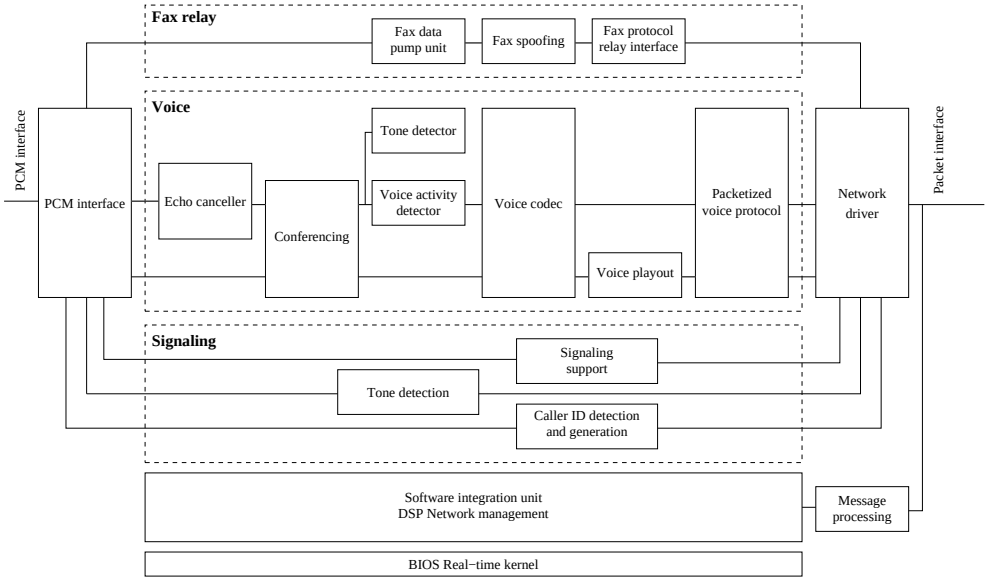
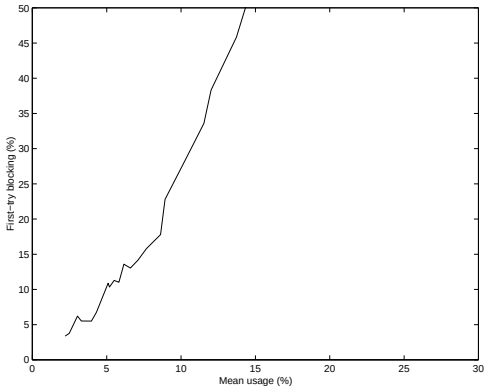


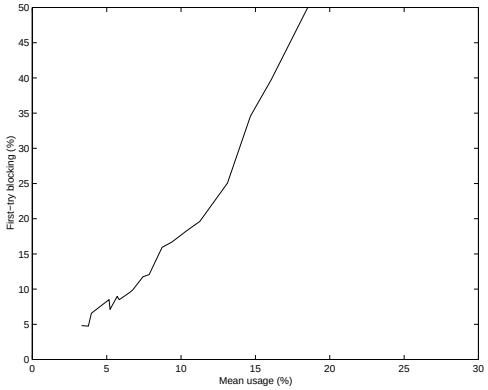
Figure 6. Software architecture for a PSTN/VoIP gateway

The stimuli simply consists of a set of transfers from a random source to a random destination at a random time with a random length where the random numbers have uniform distribution over specific intervals. This random communication patterns do not consider whether the destination port is occupied. Thus there is always a risk of the destination not being reachable at the time of transfer and can be rejected at the switch connecting to the destination port. The random pattern stimuli is mainly useful for modeling a general purpose computation platform.

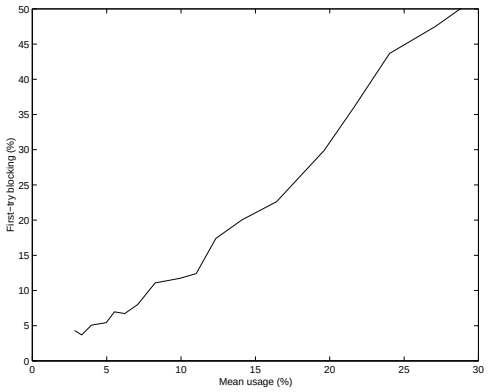
A plot of the relative first-time blocking rate vs the mean usage for a 8x8 network can be found in fig 8. The mean usage here is defined as the ratio of used payload bandwidth at the sources divided by the maximum theoretical bandwidth of the sources (i.e. 64



(a) Mean distance only



(b) Short and mean distance



(c) Short distance only

Figure 8. Relative first-time blocking ratio vs mean used payload bandwidth using random pattern stimuli

GByte/s at 1 GHz). Three different cases have been simulated. The first case, see fig 8(a), uses a completely random transfer pattern. The second case, see fig 8(b), limits the distance of half of the transfers to four hops while the second half is still random using the maximum limit of 15 hops. Finally in the third case, see fig 8(c), all the transfers are using the short distance of four hops maximum. As can be seen in the graphs, the network is not suitable for totally random traffic but can be acceptable if the traffic is mainly local and only occasionally uses longer distances. The reason is that the longer distance transfers consume more resources in the network that may be needed by other transfers and thus the probability of blocking increases.

With the random pattern simulations showing bad performance compared to the theoretical limit we propose using a pre-runtime static scheduling [14] of most communications and allowing a certain slack to allow some intermittent unscheduled traffic.

7.2. PSTN/VoIP gateway

To simulate the PSTN/VoIP gateway introduced in section 5 we assume a model that has 7x7 switches running at 1 GHz where the memory swapping ports is connected to the topmost switches and the inputs and outputs are connected to the leftmost nodes. The rest of the network is filled with 18 processors that perform a first layer of computation (e.g. echo canceling) and 18 processors that perform a second layer of computation (e.g. companding). The data flow in this simulation setup is that voice data arrives and is routed to the appropriate processor. The associated computing buffer is downloaded from the swap memory and the processing is done. When the computations have finished, the computing buffer is swapped to memory and the output from the processor is routed to the second layer processor. The second layer processor does basically the same thing and its output is sent to the correct output.

With a basic frame for a voice transfer of 4 ms that contains 32 bytes of data in conjunction with all swap buffers being 200 bytes each simulations have shown that the network capacity is up to 48000 voice channels with 0% blocking when consuming a total bandwidth of 12.8 GBytes per second if the network accesses are scheduled appropriately. The limit here is the port usage of sources and destinations that saturates before the network bandwidth is used up. Also this would require that a processor can sustain computing one voice channel in 1.5 us which is possible with the high end processors available today. The real limiting factor in such a gateway chip would most likely be the

power consumption of the IP blocks.

This limit of 48000 channels assume perfect scheduling of transfers which is not feasible in a real product. Even with significantly worse scheduling it would be possible to run a chip that handles 5000 channels together with a certain level (3-5%) of communications that are not scheduled. This is probably the real-life limit of a real gateway product based on issues such as risk of failure and economy.

8. Future work

The simulator is finished and will be used to get more simulations to prove the concept for a broader set of real-time applications. Further work is currently being done on the implementation of the SoCBUS and the goal is to build a demonstrator system that uses the SoCBUS concept. Currently the switching nodes are being implemented together with some example wrappers covering a few common interfaces. We further look into the verification of the system and the impact on system integration.

9. Conclusions

A two dimensional network for on-chip communications, the SoCBUS has been introduced. The operational principle of the SoCBUS using packet connected circuit, PCC, has been analyzed and explained in detail.

We have shown with simulations that the system is not suitable for general purpose computation platforms that exhibit random traffic patterns because of the high probability of route blocking when running without schedule. We also show that limiting the distance of communication in the random pattern case will allow a much higher bandwidth usage without saturation.

We further show that a pre-runtime static scheduling of communications approach is necessary for a hard real-time embedded system to be able to use a significant percentage of the theoretical maximum bandwidth. Using this approach we show the appropriateness for one particular application, a PSTN/VoIP gateway capable of handling up to 48000 voice channels simultaneously with perfect scheduling and that the SoCBUS will be capable of supporting a level of 5000 voice channels even with realistic scheduling and some unscheduled traffic.

References

[1] ARM, Inc., "AMBA bus description," *http://www.arm.com*.

[2] Daniel Wiklund and Dake Liu, "Switched interconnect for system-on-a-chip designs," in *Proc of the IP2000 Europe conference*, 2000.

[3] Iikka Saastamoinen, , and , "Interconnect IP node for future system-on-chip designs," in *IEEE int'l workshop on Electronic design, Test, and Applications*, 2002.

[4] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S. Malik, J. Rabaey, and A. Sangiovanni-Vincentelli, "Addressing the system-on-chip interconnect woes through communication-based design," in *Proc of the Design Automation Conference (DAC)*, 2001.

[5] "IEC/ISO 7498-1 information technology - open systems interconnection - basic reference model: The basic model," *http://www.iso.org*.

[6] Hussein G. Badr and Sunil Podar, "An optimal shortest-path routing policy for network computers with regular mesh-connected topologies," *IEEE transactions on computers*, vol. 38, no. 10, 1989.

[7] Christopher J. Glass and Lionel M. Ni, "Fault-tolerant wormhole routing in meshes without virtual channels," *IEEE transactions on parallel and distributed systems*, vol. 7, no. 6, 1996.

[8] Pierre Guerrier and Alain Greiner, "A generic architecture for on-chip packet-switched interconnections," in *Proc of the Design and Test in Europe (DATE) conference*, 2000.

[9] Daniel Wiklund and Dake Liu, "Design of a system-on-chip switched network and its design support," in *Proc of the Int'l conference on communications, circuits and systems (ICCCAS)*, 2002.

[10] William J. Dally and Brian Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc of the Design Automation Conference (DAC)*, 2001.

[11] Christer Svensson, "Optimum voltage swing on on-chip and off-chip interconnect," *IEEE Jounal of Solid-State Circuits*, vol. 36, no. 7, pp. 1108–1112.

[12] Peter Caputa and Christer Svensson, "Low power, low latency global interconnect," in *Proc of the 15th int'l ASIC/SoC conference*, 2002.

[13] William E. Witowsky and Dennis R. Gatens, "Carrier class, high density VoP," *Telogy Networks whitepaper, http://www.telogy.com*, 2002.

[14] Jia Xu, "A comparison of priority scheduling and pre-run-time scheduling," in *Proc of the Embedded Systems Conference in China*, 2002.