Bounded Arbitration Algorithm for QoS-Supported On-chip Communication

Mohammad Abdullah Al Faruque, Gereon Weiss and Joerg Henkel
University of Karlsruhe
Computer Science Department
76131 Karlsruhe, Germany
{alfaruque,weiss,henkel}@informatik.uni-karlsruhe.de

ABSTRACT

Time-critical multi-processor systems require guaranteed services in terms of throughput, bandwidth etc. in order to comply to hard real-time constraints. However, guaranteed-service schemes suffer from low resource utilization.

To the best of our knowledge, we are presenting the first approach for on-chip communication that provides a high resource utilization under a transaction-specific, flexible (i.e. different classifications on data exchange) communication scheme. It does provide tight time-related guarantees. Hence, we are presenting our bounded arbitration scheme considering lower and upper bounds for each type of transaction level. We demonstrate its advantages by means of a complete MPEG4 decoder case study and achieve under these constraints a bandwidth utilization of up to 100%, on an average 97% with a guaranteed (100%) bandwidth.

Categories and Subject Descriptors: C.3[Special-purpose and application-based systems]: Real-time and embedded systems

General Terms: Algorithms, Design

Keywords: Networks-on-chips, Bounded Arbitration Algorithm, Quality of Services

1. INTRODUCTION AND RELATED WORK

Networks on chip (NoC) gain interest since Moore's Law allows for integrating virtually hundreds of processors on a single silicon die. Efficient on-chip communication architectures will drive the future of System on Chip (SoC) architectures. One of the major challenges in NoC design is the large design space spanned by an extensive set of (partly) independent parameters. Only a set of carefully adapted parameters will allow unveiling the potential benefits of a NoC. The adapted parameters have to be integrated with the Quality of Service (QoS) requirements for designing a time-critical embedded system.

The definition of QoS is extremely diverse and application specific in embedded systems[1]. The requirements of an embedded system application (i.e.TV broadcasting in a PDA) is time critical and driven by the current work load on that device. Those quality limits are also restricted by the audio and visual perception of the human senses. The underlying QoS parameters latency, throughput and jitter in such performance phenomena, bounds must be provided by the system. Other sort of services like security data and memory read/write or Internet browsing may have varying quality requirements.

The trend of NoC in MPSoC has been recognized in the beginning of this century, where the needs and potentials of NoC as a future on-chip interconnect have been introduced [5, 7, 9]. In [12, 16], key research problems in NoC design are formalized. The issues of QoS have already been addressed in the **Previous Works** [6, 11]. But to meet these issues, with minimum overhead is still an open research challenge. Formally, on chip QoS in communication can be broadly classified into two categories: (I) *Time related performance guarantee* (i.e. bandwidth, latency and jitter) and (II) *Data-flow related reliability guarantee* (i.e. in-order data transmission, lossless data transmission). The early novel works [3, 10, 14] have partly or fully ignored the time related QoS issues. In [3] data-flow related guarantees were emphasized. To differentiate among traffic requirements, *service classes/levels* have been introduced. The service levels in a system for different communication pairs have been roughly classified in [6] as: Best-Effort (BE) service and Guaranteed Service (GS). BE-Oriented NoCs provide a higher resource utilization and treat all the communicating pairs equally and thus ignore the hard real time guarantees. In GS-oriented NoC, the research practice to provide time related guarantees are classified as (1) Establishing a connection prior to the communication and reserving resources and then sharing time slots among multiple available connections (i.e. Æthereal)[6]. (2) Implementing different service classes and exploring the advantages of wormhole routing and Virtual Channels (VCs) (i.e. QNoC)[4].

The connection-oriented technique enjoys the contention-free routing by reserving the resources in advance. Inefficient resource reservation leads to unused bandwidth. The network performs highly underutilized in Variable-Bit-Rates (VBR) transactions. The connection setup stage is an overhead for such architectures and thus, they suffer from lack of scalability. The second method (2) performs and implements quality on top of packet-switched communication, generally using the wormhole routing scheme. It provides better performance results for VBR [8], provides higher relative guarantees, but may fail in tight requirements. The problem also lies in the specification and granularity of the service classes which leads to unused bandwidth. These approaches use a simple *Round Robin* link arbitration shown in Algorithm 1.

To the best of our knowledge **our approach** is the first one, that considers both of these two approaches to provide QoS. It has adopted the fine-granular service class specification considering the lower and upper bounds for each transaction type, thus avoids resource underutilization and erroneous resource reservation. On top of approach (2) it

[1]Embedded system and time-critical system is used interchangeably in the scope of this paper

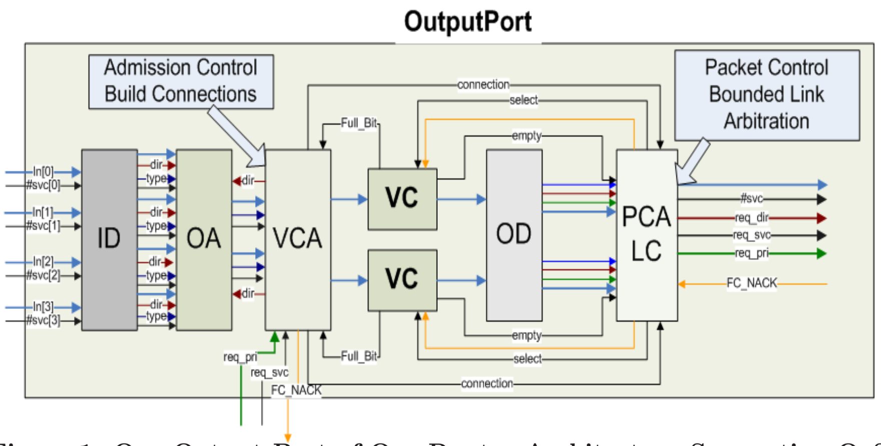


Figure 1: One Output Port of Our Router Architecture Supporting QoS.

provides flexible connections for the required service classes. It implements the dynamic *scheduling table* and flexible link arbitration algorithm, which is aware of the present traffic load, our *Bounded Arbitration Algorithm*.

The work presented in [13] has assumed that all the VCs are statically allocated and predetermined by the ***Configuration Manager (CM)***, a central authority. QNoC provides on an average slightly more than 99% guarantee but fails to achieve 100% guarantee, which is a must condition for a safety critical embedded system. The work presented in [4] and [8] implement arbitrations depending on the service classes. They provide *relative guarantee*, no *tight guarantees* compared to their whole architectures. Common to all current work in NoC is that efficient resource utilization together with supporting hard guarantees is not fully considered.

The rest of the paper is organized as follows. In Section 2, we present our novel contribution. In Section 3, we introduce our QoS-Supported NoC. In Section 4, we introduce our hardware implementation. Experimental results are presented in Section 5, and we finally conclude in Section 6.

2. OUR NOVEL CONTRIBUTION

Our novel contributions are as follows:

(1) We have designed a new arbitration technique, BAA concerning the lower and the upper bound of individual transaction type. The arbitration is dynamic in nature depending on the current traffic load but ensures the bounds and suits the traffic characteristics.

(2)The problem of fine granular bandwidth requirements to save unused bandwidth is also handled with fine granular service class specification.

(3) The distributed programming model to establish a guaranteed connection on top of a fine granular service class is also accomplished.

Hence, our NoC provides tight time-related guarantees by our link arbitration process and maximizes the link utilization. Besides these unique characteristics, our approach does not incur any performance penalties. In order to evaluate the claims of our new link arbitration algorithm, we have presented a case study analysis with real world application, an MPEG4 video decoder [2]. We achieved as high as 100% and on an average 97% bandwidth utilization, not achieved so far.

3. QOS-SUPPORTED NOC DESIGN

The goals of our NoC project are twofold: (1) A concept for guaranteeing QoS of a NoC: guaranteed bandwidth, low latency and jitter; (2) A rapid prototyping environment for NoCs that allows specification and evaluation of a customized NoC within hours through a proprietary NoC IP library. As a result, cycle-accurate transaction-level data is obtained through execution on an FPGA platform, allowing for fast design space exploration. The advantage of virtual channels in the wormhole routing scheme together with priorities, as fine grained service class specification are used to provide per-connection guarantees on top of a packet-switched network. Dynamic channel preemption and establishment is used for some highly prioritized connections that are part of, for example, a security task.

Table 1: Fine-grained Service Class Specification

Services	Latency	BW	Jitter	Example
Prio H	High	-	-	short message
Prio X1	- -	Fixed guarantee	No jitter/ fixed	streaming video 80%BW
Prio X2	- -	Fixed guarantee	jitter allowed	streaming data 50%BW
... ...	... ...	... ...	... ...	... ...
Prio X_n	- -	Fixed guarantee	No jitter No jitter	streaming aud io high latency
Prio L	no limit	Not fixed	accep table	short memory access

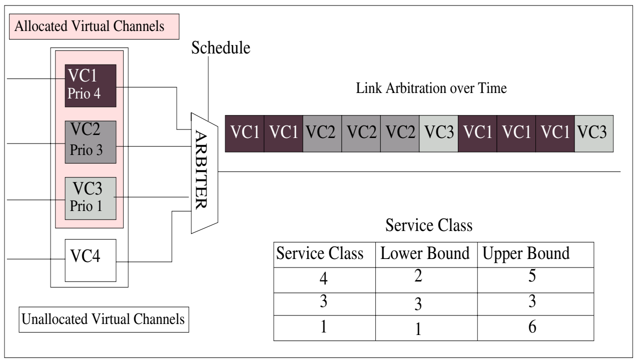
Algorithm 1 RR: ***Round Robin link arbitration***

```
Number of virtual channels: VC
Number of reserved virtual channels at time t : $VC_{res}$
$i^{th}$ entry in the Slot Table: ST(i)
0. for each reserved virtual channel $i = 1$ To $VC_{res}$ {
1.     $ST(i) = i$
2.     $++next\_slot$
3. }
```

3.1 Assumptions in our communication Scheme

The NoC architecture needs to be simple and configurable in structure. Our architecture is pipelined like the state-of-the-art Xpipe architecture [3]. We have used wormhole routing because of its low latency in practice and small buffer space requirements [15]. Source based deterministic routing is used in the current architecture. The routing algorithm can be configured from any of the deadlock free deterministic routing algorithms. In the current implementation, XY routing is used. The topology is kept as a grid based structure. The links between two routers are pipelined, so, irregular topology with a favorable routing algorithm can also be designed within the frame-work.

To keep the NoC predictable in terms of cycle counts, the router-to-router connections and also the router-to-IP connections are cycle-accurate. To demonstrate the NoC architecture, one output port unit for a simple router is shown in Figure 1. The *Input Decoder (ID)* decodes the incoming packets for the appropriate output port. *Output Arbiter (OA)* arbitrates the output directions among the incoming packets. Then the following steps implement the QoS during the transmission. To provide the service class ***H*** (the highest priority explained in the next subsection), easy access at anytime, a constraint is kept in the NoC architecture. If the


Allocated Virtual Channels
Schedule
VC1
Prio 4
VC2
Prio 3
VC3
Prio 1
VC4
ARBITER
Link Arbitration over Time
VC1 VC1 VC2 VC2 VC2 VC3 VC1 VC1 VC1 VC3
Unallocated Virtual Channels
Service Class

Service Class	Lower Bound	Upper Bound
4	2	5
3	3	3
1	1	6

Figure 2: Bounded Link Arbitration.

number of virtual channels are ***N***, then there can be only ***N-1*** number of fixed bandwidth related connections. Fixed bandwidth related connections need to be kept because of tight guarantees. To allow a higher operating frequency, the router structure is deeply pipelined. The output port is pipelined to 6 pipeline stages. No error control unit is added in the architecture. The network is assumed to be reliable. Hence, no end-to-end retransmission is required and thus, the throughput and the latency bounds can be measured cycle accurately. The two necessary stages for a guaranteed connection, the *admission control stage* and the *packet control stage* are also highlighted in the figure. The *Virtual Channel Arbiter (VCA)* implements the admission control and the *Physical Channel Arbiter/Link Arbiter (PCA/LA)* handles the new link arbitration algorithm.

Algorithm 2 Priority-Weighted Round Robin

```
Priority of the $i^{th}$ connection : $P_i$
Cycles assigned to priority P : $C_p$
0.  for each reserved virtual channel $i = 1$ To $VC_{res}$ {
1.        for $j = 1$ To $C_p$ {
2.          $ST(i) = i$
3.          $++next\_slot$
4.        }
5. }
```

3.2 Fine-grained QoS Specification

The support of diverse types of traffic patterns with the maximum link utilization is the principal goal of quality of service oriented NoC design. In the scope of this paper, support for guaranteed bandwidth, low latency and jitter and a scalable frame work for the service class assignment are specified. Previous work like [6] has summarized the service classes in: *best effort* and *guaranteed throughput*. Work in [4] has identified 4 different types of service classes: *Signaling*, *Real-Time*, *Read-Write (RD/WR)* and *Block Transfer*. All sort of real time traffic is classified as service class *Real-Time*. So for all real time connections with varying communication requirements, (i.e. bandwidth) they treat the connection in the same fashion. However, all these works underestimate the fine granularity of the bandwidth assignment. To overcome these shortcomings, our approach offers a scalable and application specific service class specification. In Table 1, a typical service class specification of our approach is shown. Each of the service classes specifies lower and upper bound cycles in the scheduling table[2] to satisfy VBR data transactions. The jitter problem is handled by keeping the lower and upper bound the same. The highest priority is kept for the messages having emergency requirements, i.e. interrupts, security messages.

3.3 Our Bounded Arbitration Algorithm (BAA)

Considering all time-related requirements, in Algorithm 2, our first approach with *Prioritized Round Robin* is depicted[3]. This algorithm provides service class specification as well as it provides tight guarantees in terms of bandwidth and jitter. *Priority-Weighted Round Robin* exploits the advantage of Time Division Multiplexing (TDMA) together with an independent service class specification. Each type of service classes gets a fixed number of cycles to send data at a specified time on the link. The diversity of data transactions under different traffic loads is not taken into account. It can not configure services adaptively while meeting the lower bound. This arbitration scheme would not adapt to different quality of service requirements like the motivational example given in the introductory chapter. These situations may lead to false reservation according to miss-use of available resources during average execution time. In the presence of available bandwidth all the bandwidth can be assigned to the lowest priority (like Æthereal)[6]), best effort traffic. But a situation of having no available best effort traffic will lead to bandwidth underutilization for the higher prioritized traffic.

Algorithm 3 Bounded Arbitration Algorithm

```
Slot table size in number of cycles : $ST_{max}$
Slot available in number of cycles : $ST_{avl}$
currently assigned cycles to virtual channel k: $SLOT_k$
VC k latency sensitive and jitter allowed: $VC_k^{delay\_jitter}$
VC k jitter allowed: $VC_k^{jitter}$
Upper bound # of cycles assigned to priority p : $C_{max}^p$
Lower bound # of cycles assigned to priority p: $C_{min}^p$
1.  slot_assigned=true
// Fill Scheduling Table(SC) W/ lower bound//
2.  for each reserved virtual channel $i = 1$ To $VC_{res}$ {
3.    for $j = 1$ To $C_{min}^p$ {
4.       $ST(next\_slot) = i$
5.       $++next\_slot$
6.    }
// Filling SC W/latency sens. VC, allows jitter//
7.    while slot_assigned {
8.       $slot\_assigned = false$
9.       for $k = 1$ To $VC_k^{delay\_jitter}$ {
10.         if ($SLOT_k < C_{max}^p$ AND $ST_{avl}$) then {
11.           $ST(next\_slot) = k$
12.           $++next\_slot$
13.           $++SLOT_k$
14.           $slot\_assigned = true$
15.         }
16.       }
17.   }
// Filling SC W/ VC, allows jitter (rest BE)//
18.   $slot\_assigned = true$
19.   while slot_assigned {
20.      $slot\_assigned = false$
21.      for $k = 1$ To $VC_k^{jitter}$ {
22.        if ($SLOT_k < C_{max}^p$ AND $ST_{avl}$) then {
23.          $ST(next\_slot) = k$
24.          $++next\_slot$
25.          $++SLOT_k$
26.          $slot\_assigned = true$
27.        }
28.      }
29.   }
```

The latency and the resource underutilization problem is considered in Algorithm 3. To avoid the latency problem and to use the unused bandwidth by any idle connection we are proposing **our novel algorithm**, a link arbitration algorithm, called *Bounded Arbitration Algorithm (BAA)*. Unused bandwidth can be shared by every present communication level connection, but still considering the service class specification. This tends to increase throughput for the connections. The less time a resource is (exclusively) reserved the chance of contention for the resources by the concurrent connections decreases. Thus, the algorithm is not only

[2]we use interchangeably scheduling table and slot table
[3]Previously assigned variables in the algorithms are reused

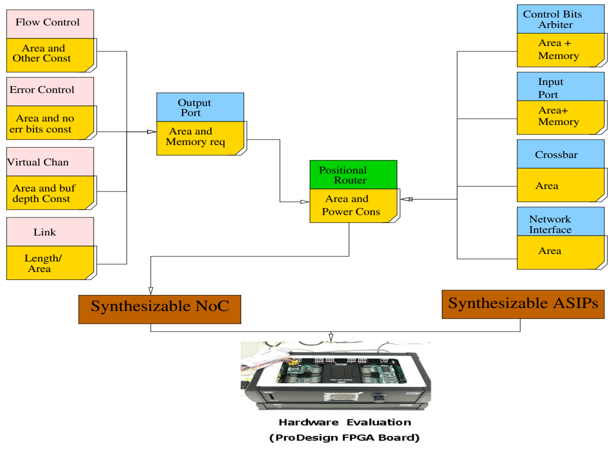


Figure 3: FPGA Prototyping of a NoC.

providing low latency for privileged connections but also a possible better utilization for other connections/applications using the NoC. The algorithm is implementing the service class specification together with the higher and lower bound slot allocation. In Figure 2 and in Algorithm 3, the new link arbitration algorithm, the bounded arbitration algorithm is shown. In this algorithm scheme, all connections first meet their lower bound requirements and then go for filling the unused bandwidth approaching to the upper bound. The selection of the virtual channels up to the upper bound, specified by the priority of the stored packets is done depending on the service class parameters, specially considering latency and jitter. The algorithm ensures the lower bound for any sort of transaction but surely on the service class basis. But besides, offering unused bandwidth to only lower priority connections, it tends to meet the upper bound requirements for some flexible transactions.

3.4 Bound Analysis

Building a time-critical embedded system is always a challenging task. The system design starts from the application exploration. This exploration provides the average, lower and best case scenario for the running system. From this analysis the designer restricts the design to obey the bounds. We can formulate the throughput bounds (in bits/cycle) for every connection as:

$$[L\text{-}B]\ T_c^p = (low\_s_c^p/S) \times CW \times DR\ [bits/cycle] \quad (1)$$

$$[U\text{-}B]\ T_c^p = (up\_s_c^p/S) \times CW \times DR\ [bits/cycle] \quad (2)$$

The throughput for a connection $\boldsymbol{c}$ having the priority $\boldsymbol{p}$ is defined by T_c^p in the above equation. The number of allocated cycles for the connection $\boldsymbol{c}$ over time for priority $\boldsymbol{p}$ is assigned both for lower and upper bounds, $low\_s_c^p$ and $up\_s_c^p$. The symbol CW stands for the channel width and DR stands for the data ratio, representing the protocol overhead.

Similarly, the latency bounds can be summarized as:

$$L_{hop}^i = L_{vctolinkS}^i + L_{link}^i + L_{linktoVC_j}^i \quad (3)$$

$$L_c^p = (L_{IPtoSW} + L_{hop}^i + L_{NItoIP}) \times IP\_data/T_c^p \quad (4)$$

Here, L_{hop}^i is the latency of an individual hop $\boldsymbol{i}$. The latency in an individual hop is calculated by summing up the latency from the virtual channel to the link ($L_{vctolinkS}^i$), the latency in the link (L_{link}^i) and the latency for the router $\boldsymbol{i}$ in link to the virtual channel $\boldsymbol{j}$ ($L_{linktoVC_j}^i$). After attaining the latency in a single hop, we can calculate the latency for an individual transaction. Symbol L_c^p stands for latency of a connection having the priority $\boldsymbol{p}$. This latency can be calculated by all the small parts that contribute to the latency along the path, the latency from the IP to the router switch

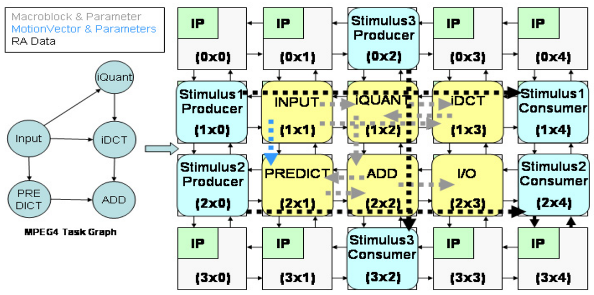


Figure 4: MPEG4 Mapping in a 5×4 Mesh.

(L_{IPtoSW}), the latency for each router $\boldsymbol{i}$ in the connection path (L_{hop}^i), the latency from the NI to the IP (L_{NItoIP}). All these sources of latency are multiplied with the amount of data sent by the IP (IP_{data}) and divided by the throughput of the connection having the priority $\boldsymbol{p}$ (T_c^p).

Our quality of service oriented NoC needs to be programmed efficiently. Two possible programming models available in NoC literatures are: programming the NoC having the global view of the whole NoC, a *centralized approach* and no global view a *distributed approach*. In our NoC, we have followed the distributed approach. The dynamic connection establishment and channel preemption is adopted in our scheme. Low priority packets can release the channel only after the tail flit is sent. The new header flit with a higher priority has to wait for that time interval.

4. HARDWARE IMPLEMENTATION

Our NoC provides a flexible service class specification together with configurable QoS issues. Considering all the configurable parameters our approach can generate a synthesizable NoC specification through a VHDL and SystemC based proprietary NoC IP library. On the upper abstraction level it allows fast HW/SW co-simulation, on the lower abstraction level it provides rapid prototyping on a scalable FPGA platform, a ProDesign board [1]. The synthesizable NoC is build from our NoC component IP library. In Figure 3, a schematic diagram of the FPGA prototype of a NoC is shown. The synthesizable library is highlighted in the figure. It comprises a collection of *positional routers* depending on the topology. [4] Each positional router is built using IP blocks: Output-port, Input-port, Network Interface (NI), Crossbar and Arbiter. Output queuing is used throughout all design alternatives. Each of the output ports can be designed using the following components: Virtual Channels, Links and Flow Control logic. All of these network components are freely parameterizable.

Our BAA algorithm comes with a little fixed hardware overhead. The service class specification with the attributes of lower and upper bound has to be hard-coded in each router. The service class table size is linear to the fine-grained specification. The more the number of service classes the bigger is the *service class table*. But the table is constant in size from the design perspective. The length of the *scheduling table* also depends on the number of virtual channels: if the number of virtual channels is n where $n \approx 2^m$, then m bits are required to encode each slot in a scheduling table. The scheduling table size (number of slots in a scheduling table) is kept equal to the size of the lower bound slot requirements of the highest prioritized connections. For example, if there are k virtual channels then there can be *(k-1)* connections with bandwidth and jitter sensitivity. One connection can be used for higher H or any other flexible

[4] The term *positional router* is used in this paper to highlight the need of different orientation of input and output ports.

Table 2: Traffic Modeling

Injection:	Min.	Max.
MPEG4	400 MB/s	800 MB/s
STIMULUS1	120 MB/s	400 MB/s
STIMULUS2	180 MB/s	400 MB/s
STIMULUS3	300 MB/s	600 MB/s

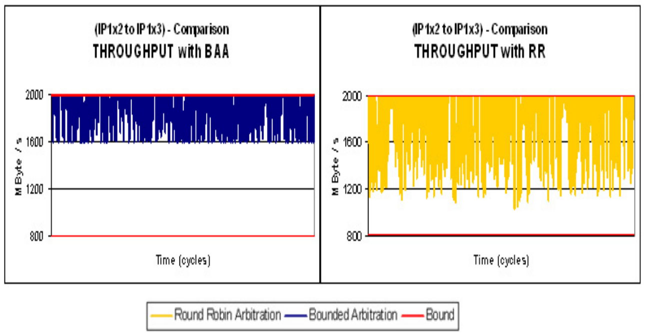


Figure 5: Throughput between RR and BAA.

service class. So the scheduling table size, T_{size} is,

$$T_{size} = (m - 1) \times S_{band-sensi} + S_H \quad (5)$$

Here, $S_{band-sensi}$ is lower bound slot of highest prioritized bandwidth sensitive connections and S_H is lower bound slot of the service class H. The resulting array is stored in a FPGA Block RAM. The biggest part of the area comes from the output buffers in each router. The minimum size of the buffer is determined by the depth of the pipelining on the link because of the hop-to-hop flow control. If the links are pipelined into x stages (1 cycle for each stage) and processing time in pair routers take y cycles then the minimum number of unit buffer in flit size is (2x + y). This is the given lower bound as in [3]. The buffer for each router increases linearly with the introduction of an additional virtual channel in each router output port.

5. RESULTS AND MPEG4 CASE STUDY

For the evaluation of the design and runtime adaption of **our NoC** frame work we implemented an MPEG4 video decoder. An MPEG4 encoded video is a sequence of *I-frames*, *P-frames* and *B-frames*. An I-frame is not predicted. P-frames are predicted from the previous I- or P-frame. And finally, a B-frame is predicted from the previous and next I- or P-frame. B-frames are optional in compression. I-frame needs the tasks, *INPUT*, *iQuant*, *iDCT* and *ADD* for decoding. P- and B-frames require *INPUT/VLC*, *iQuant*, *iDCT*, *PREDICT* and *ADD* for decoding. Our mapping of an MPEG4 video decoder to the 6 inner tiles of a 5×4 Mesh NoC Architecture is shown in Figure 4. Other tiles are used for other ***Stimuli*** to create arbitrary traffic during the experiments. It simulates the embedding of the MPEG4 in a larger application. The smallest communication units between two tasks are also shown in the picture. In most of the cases the smallest communication unit in MPEG4 is a *MacroBlock(MB)*, 16×16 pixels. The communication unit between the tasks INPUT and PREDICT is the so called *MotionVector(MV)*.

$$\alpha(t) = \mu = \frac{\rho\prime(t) \times D}{\rho\prime\prime(t) \times W} \quad (6)$$

For the experiments, we have taken the traffic model described in Table 2 and in Equation 6. A random rate, μ keeps the injected traffic load between 30-100% of the worst case specified in the Table 2. Here D stands for the amount of data transferred between two tasks and W for the

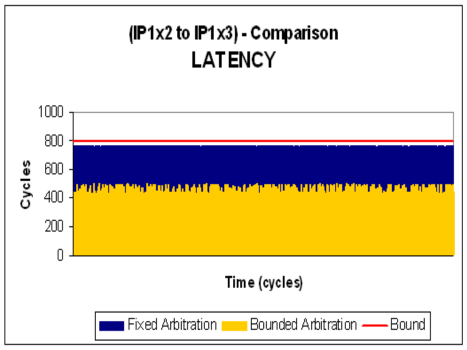


Figure 6: Latency for Fixed and BAA.

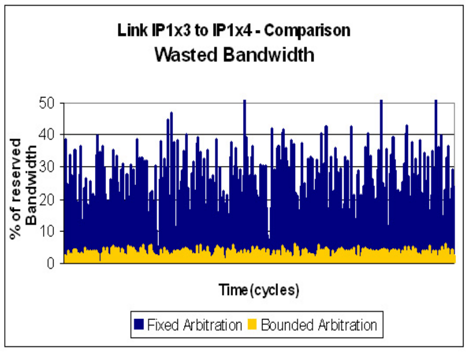


Figure 7: Resource Utilization: Fixed and BAA.

worst case (fastest computation) execution time. To keep the traffic application-specific, random variables $\rho\prime$ and $\rho\prime\prime$ have been introduced, where $0 \leq \rho\prime \leq 1$ and $1 \leq \rho\prime\prime \leq 2$. Experiments show, the injection rates of VBR data producing applications vary over time. For Example, the MPEG4 IP1x2 (iQuant) is producing a MacroBlock of 808 Bytes for IP1×3 (iDCT) in a interval depending on the prior processing of IP1x1 (Input). Our observations of the MPEG4 application show, the computation times vary from 50-100% of the worst case time. Some observations considering the connections iQuant → iDCT and STIMULUS1(Producer)→ STIMULUS1(Consumer) are recorded in the following given graphs.

Figure 5 shows that RR is separating different connections, but only fair by the meaning of equality. Every connection is receiving the same bandwidth. In the current scenario STIMULUS1 transaction is not latency-sensitive and only needs less bandwidth than the MPEG4 connection. But even then STIMULUS1 is obtaining the half of the bandwidth, while both connections are sending traffic. No traffic classification can be done for the connections in the RR scheme [13]. An interesting graph is the throughput for MPEG4, where the bounded arbitration is keeping the throughput very high up to the maximum bandwidth for the latency-sensitive traffic. The RR arbitration is really fluctuating the bandwidth between the half and maximum bandwidth. This fluctuation would increase with every additional connection along the path. Thus the average throughput for the latency-sensitive MPEG4 for the RR algorithm will be low compared to the BAA.

Figure 6 shows the difference between the fixed arbitration (TDMA like), sharing reserved bandwidth among connections, with the BAA. Every static TDMA based connection-oriented architecture has to deal with reserved and not used resources because of the unpredictability of complex applications with VBR transactions. Æthereal like architectures can only use BE-service for already reserved resources. The problem here is: the amount of BE-traffic of concurrent applications has to be high enough to compensate. The Æthereal architecture is thus, designed for underutilization for the ability to give hard guarantees. A connection can be han-

Table 3: Latency, Throughput and Utilization (RR and BAA looks the same but BAA has better latency and throughput results) comparison (P/B frames for MPEG4)

	Average Throughput MB/s			Average Latency Cycles/1000b			% Bandwidth Waste		
Transactions	Fixed	RR	**Our BAA**	Fixed	RR	**Our BAA**	Fixed	RR	**Our BAA**
Input− >iQuant	800	1840	**1901**	939	575	**561**	90	0	**0**
Input− >iDCT	800	1822	**1900**	1272	883	**850**	83	1	**1**
Input− >PREDICT	800	2000	**2000**	1680	986	**978**	89	2	**2**
iQuant− >iDCT	800	1756	**1875**	962	599	**573**	90	0	**0**
iDCT− >ADD	800	1662	**1767**	3148	1620	**1499**	57	1	**1**
PREDICT− >ADD	800	1768	**1890**	1051	626	**590**	55	1	**1**
STIMULUS1	400	1788	**1732**	1754	676	**709**	23	4	**8**
STIMULUS2	400	1730	**1673**	1717	703	**739**	25	4	**4**
STIMULUS3	600	1777	**1737**	1236	648	**668**	19	3	**4**

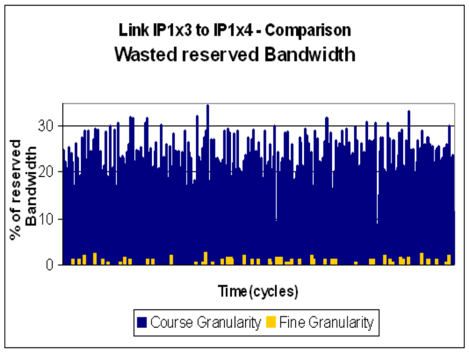


Figure 8: Fine-granular Service Class Spec.

dled fairly among the others because of its service class. The MPEG4 application is always obtaining the highest bandwidth in the BAA scenario, because it is privileged by its latency sensitive specification. The left bandwidth can be used by every other connections. But the prioritization and the service classes determine the type of sharing. The observation shows that the MPEG4 application latency is lower for the BAA than for the fixed arbitration because bandwidth is reserved and not used again. The results show that the sharing of unused reserved bandwidth even among connections (not only to BE-services) is crucial for a better utilized network.

The waste of bandwidth for a static reservation of resources compared to a more adaptive, BAA, is shown in the Figure 7. The BAA is wasting less of the reserved bandwidth while the fixed approach is wasting 25% because of the unpredictability of the applications injection rate. This figure points out that the waste of bandwidth in a static approach is too high in a small network on a chip. The amount of wasted bandwidth is not acceptable for the high on-chip resource costs . So there is a mandatory need for an adaptive strategy like our BAA.

The possible granularity of the traffic classification is crucial for the utilization of the network. The STIMULUS1 is producing data with 20% of the bandwidth. One classification reserves 25%, the other one 20%. BAA is used, but with no jitter. The results in Figure 8 show that, there is on an average 20% waste of the reserved bandwidth as expected for 5% error in the classification. So the conclusion is, a fine-granular traffic classification should be used within a NoC frame-work to keep the underutilization of the network to an acceptable range. Other priority-based architectures are not offering this scalability. The advantage of the BAA in terms of higher throughput, low latency and more resource utilization compared to the fixed TDMA based approach and to the simple round robin policy is summarized in Table 3.

6. CONCLUSION

In this paper, we have introduced the importance of classifying each transaction level at its granular level with lower and upper bounds of services. A scalable on-chip interconnect with bandwidth guarantee is beneficial for on-chip real-time multi-processor systems. However, it carries the burden of typically wasting a large amount of bandwidth when the guaranteed bandwidth is not used by one or more interconnect channels. As discussed, state-of-the-art does not provide satisfactory solutions. Either they lack 100% guarantee or they come with a relatively large average bandwidth waste.

We have demonstrated that our scheme is actually able to provide a 100% guarantee of bandwidth with an average waste of only 3% (i.e. 97% utilization) a value that has not been achieved by others so far. We have evaluated our scheme by a thorough case study of an MPEG4 decoder under varying stimuli scenarios. Besides, we have introduced our on-chip interconnect platform.

7. REFERENCES

[1] http://www.prodesigncad.com.
[2] http://www.xvid.org/.
[3] L. Benini and G. D. Micheli. Networks on chips: A new soc paradigm. *Proc. of IEEE Computer,35(1)*, 2002.
[4] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny. QNoC: QoS architecture and design process for network on chip. *Special issue on Networks on Chip, The Journal of Systems Architecture*.
[5] W. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. *Proc. of the Design, Automation conf.(DAC'01)*, June. 2001.
[6] E. Rijpkema et. al. Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip. *Proc. of the Design, Automation and Test in Europe conf.(DATE)*, March. 2003.
[7] P. Guerrier and A. Greiner. A generic architecture for on-chip packet switched interconnections. *Proc. of the Design, Automation and Test in Europe conf.(DATE)*, March. 2000.
[8] M. D. Harmanci, N. P. Escudero, Y. Leblebici, and P. Ienne. Quantitative modelling and comparison of communication schemes to guarantee quality-of-service in networks-on-chip. *International Symposium on Low Power Electronics and Design.*
[9] R. Ho, K. Mai, and M. Horowitz. The future of wires. *Proceedings of the IEEE,pages 490-504*, April. 2001.
[10] I.Saastamoinen, D.Siguenza-Tortosa, and J. Nurmi. Interconnect IP node for future system-on-chip designs. *Proc. of The First IEEE International Workshop on Electronic Design, Test and Applications*, pages 116–120, January. 2002.
[11] J. Henkel et. al. On-chip networks: A scalable communication-centric embedded system design paradigm. *Int. Conf. VLSI Design*, 2004.
[12] A. Jantsch and H. T. (Eds). Networks-on-chip. *Kluwer.*, 2003.
[13] N. Kavaldjiev, G. J. M. Smit, P. G. Jansen, and P. T. Wolkotte. A virtual channel Network-on-Chip for GT and BE traffic. *Proc. of the International Symposium on VLSI (ISVLSI'06).*
[14] S. Murali and G. D. Micheli. Sunmap: A tool for automatic topology selection and generation for nocs. *Proc. of the Design, Automation conf.(DAC'04)*, June. 2004.
[15] L. Ni and P. McKinley. A survey of wormhole routing techniques in direct networks. *IEEE Computer*, pages 62–75, February. 1993.
[16] U. Y. Ogras, J. Hu, and R. Marculescu. Key research problems in noc design: A holistic perspective. *CODES+ISSS*, September. 2005.