Microprocessors and Microsystems 37 (2013) 333–344

Contents lists available at SciVerse ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Traffic engineered NoC for streaming applications

Basavaraj Talwar *, Bharadwaj Amrutur

Electrical Communication Engineering Department, Indian Institute of Science, Bangalore 560 012, India

ARTICLE INFO

Article history:
Available online 19 February 2013

Keywords:
Networks-on-chip
Chip-multiprocessors
QoS
Label switching
Bandwidth reservation
Streaming applications

ABSTRACT

Streaming applications demand hard bandwidth and throughput guarantees in a multiprocessor environment amidst resource competing processes. We present a Label Switching based Network-on-Chip (LS-NoC) motivated by throughput guarantees offered by bandwidth reservation. Label switching is a packet relaying technique in which individual packets carry route information in the form of labels. A centralized LS-NoC Management framework engineers traffic into Quality of Service (QoS) guaranteed routes. LS-NoC caters to the requirements of streaming applications where communication channels are fixed over the lifetime of the application. The proposed NoC framework inherently supports heterogeneous and ad hoc system-on-chips. The LS-NoC can be used in conjunction with conventional best effort NoC as a QoS guaranteed communication network or as a replacement to the conventional NoC.

A multicast, broadcast capable label switched router for the LS-NoC has been designed. A 5 port, 256 bit data bus, 4 bit label router occupies 0.431 mm^2 in 130 nm and delivers peak bandwidth of 80 Gbits/s per link at 312.5 MHz. Bandwidth and latency guarantees of LS-NoC have been demonstrated on traffic from example streaming applications and on constant and variable bit rate traffic patterns. LS-NoC was found to have a competitive $\frac{\text{Area}\times\text{Power}}{\text{Throughput}}$ figure of merit with state-of-the-art NoCs providing QoS. Circuit switching with link sharing abilities and support for asynchronous operation make LS-NoC a desirable choice for QoS servicing in chip multiprocessors.

1. Introduction

Network-on-Chips [1–3] help manage high complexity of designing large chips by decoupling computation from communication. NoCs servicing generic chip multiprocessors (CMPs) or customized system-on-chips (SoCs) are expected to meet Quality of Service (QoS) demands of executing applications. The two basic approaches in NoC designs to enable QoS guarantees are: creation of reserved connections between source and destinations via circuit switching or support for prioritized routing (in case of packet switched, connectionless paths). Packet switched networks provide efficient interconnect utilization and high throughputs [4]. However, they need to be over-provisioned to support QoS for various traffic classes and have high buffer requirements in routers. On the other hand, circuit switched NoCs guarantee high data transfer rates in an energy efficient manner by reducing intra-route data storage [5]. These are well suited for streaming applications where communication requirements are well known.

Streaming applications such as HiperLAN/2 Baseband Processors [6], Real-time Object Recognition Processors [7] and H.264 encoders [8,9] have well understood communication patterns and bandwidth requirements. Adequate throughput, latency and bandwidth guarantees between process blocks can be provided by establishing provisioned, contention-free routes between nodes.

Many existing NoC works target guaranteed services and bounded latency in CMPs and SoCs [5,6,10–14]. These methods use hardware or software managed resource reservation techniques or implement priority-based schemes in conjunction with resource sharing between connections across time and space. Section 2 provides a survey of these NoCs and techniques. Priority-based schemes reduce to best effort service when all traffic belongs to the same priority class. Priority-based schemes for QoS work best in a dynamic, large system such as the Internet. A centralized, resource reservation scheme is practically infeasible due to magnitude and complexity of such a system. On the other hand, on-chip networks are smaller and less dynamic. In CMPs, an application's processes are scheduled into individual cores by the operating system. Input data and traffic characteristics of streaming applications in a CMP environment can be easily profiled. SoCs are designed to meet specific application demands with well known input, output and traffic scenarios.

Circuit switching schemes have been shown to deliver QoS for specific applications [5,6]. Circuit switching schemes that aim at identifying shortest path circuits in a static manner block network resources for the lifetime of the connection. Such schemes result in inefficient utilization of network resources.

* Corresponding author. Tel.: +91 80 22932749.
E-mail address: bt@ece.iisc.ernet.in (B. Talwar).

0141-9331/$ - see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.micpro.2013.02.003

In this paper, we present a Label Switching based Network-on-Chip (LS-NoC) motivated by throughput guarantees offered by bandwidth reservation. Such an NoC can be used to service hard bandwidth and throughput guarantees to streaming applications in a multiprocessor environment amidst resource competing processes. The Label Switched (LS) Router used in LS-NoC achieves single-cycle traversal delay during contentionless traffic, is multicast and broadcast capable. Source nodes in the LS-NoC can work asynchronously as cycle level scheduling is not required in the LS Router.

1.1. Organization of the paper

Section 2 lists some of the existing QoS based NoC designs. Design of the LS-NoC router, pipe establishment algorithm and related LS-NoC details are presented in Section 3. Section 4 outlines LS-NoC Manager and estimates its overhead. Experiments with streaming application case studies are presented in Section 5. The paper concludes in Section 6 after enlisting some future advancements possible on the current work.

2. Related work

Providing QoS guarantees in on-chip communication networks has been identified as one of major research problems in NoCs [15]. QoS solutions in packet switched networks use priority-based services while circuit switched NoCs use some form of resource reservation. We introduce a few well known QoS solutions from literature and compare our work with the state of the art. Packet switched NoCs use differentiated services for traffic classes [7,14,16,17] to provide latency and bandwidth guarantees. Circuit switched NoCs use resource reservation mechanisms to guarantee QoS [5,10,13,18]. Resource reservation mechanisms involve identifying a sufficiently resource rich path, reserving resources along the path, configuration, actual communication and path tear down. An extensive survey of NoC proposals has been presented in [19]. Relevant QoS NoCs are discussed in this section.

2.1. QoS in packet switched networks

One of the major drawbacks of priority-based QoS schemes [14,16,17,20,21] is that increase in traffic in one priority class effects the delay on traffic belonging to other classes. A priority-based QoS network loses the differentiated services advantage if all traffic belong to the same priority level. Further, deadlock-free routing algorithms using virtual circuits with a priority approach may lead to degradation in NoC throughput.

2.2. QoS in circuit switched networks

Resource reservation methods use one of the following two methods: (a) a path probing, service network, (b) an intelligent, traffic-aware distributed or centralized manager. Circuit switched, bus-based QoS solutions such as Crossroad [22], dTDMA [23] and Heterogeneous IP Block Interconnection (HIBI) [19] rely on communication localization to satisfy timing demands.

Point-to-point connections [24] in NoCs are costly in terms of network resources and result in inefficient network utilization and poor scalability. Crossbar based solutions [12,25] use protocol handshakes to establish a communication path between nodes. Such protocol handshakes force nodes to wait till handshake is complete before establishing the path. Non-interference of communication channels is achieved by over-provisioning resources in the crossbar. This leads to complex and poorly scalable networks. Static routing along shortest paths does not guarantee latency bound routes due to arbitration delays in the network.

In cases where connections cannot be overlapped with each other (e.g. MANGO NoC [26]), increased number of hard GT connections will lead to increased cost in network resources. In the MANGO NoC, sharing of links by VCs results in increased delays in an individual VC, when the link is saturated with VCs. In LS-NoC, new pipes are configured in a contention-free manner to guarantee adequate network resources per pipe.

Intel's 8×8 circuit switched NoC [5], SoCBUS [13,27] and the distributed programming model in Æthereal [28] use probe-based circuit establishment solutions. In these NoCs, probe packets are used to reconnoiter shortest communication paths and configure routing tables if a circuit is available. Routers are locked down and no other circuits can use the port during the lifetime of an established circuit. If the shortest X–Y path is not available, the probe packets initiate route discovery mechanisms in other paths. The probe-based route discovery mechanism is dynamic and is dependent on the current traffic in the NoC. If the circuit establishment does not succeed in the first attempt, the probe packet might repeat route discovery steps or try after a random period of time. This leads to indeterministic and sometimes large route setup times which may be unacceptable for real time application performance.

2.2.1. Centralized circuit management

Reserved communication channels can be identified and configured using an application aware hardware or software entity [18,10]. Such a traffic manager can provide programmability of routes.

The Octagon NoC [18] implements a centralized, best-fit scheduler to configure and manage non-overlapping connections. Connection reservation by peer nodes delays establishment of a new connection by the scheduler. This results in increased connection establishment time at the routers and also packet losses.

The Æthereal NoC [10] aims at providing hard guaranteed QoS using Time Division Multiplexing (TDM) to avoid contention in a synchronous network. The centralized programming model in Æthereal NoC [10] uses a root process to identify free slots and configure network interfaces.

TDM techniques using slot tables in Æthereal [10] and sequencers in Adaptive System-on-Chip [11] require a single synchronous clock distributed over the chip. Accurate globally synchronous clock distribution is expensive in terms of power. Global synchronicity can be achieved in a distributed manner using tokens – such that every router synchronizes every slot with all of its neighbors [29]. The slowest router dictates the operating speed of the NoC. Further, power management techniques such as multiple clock domains is not feasible with this approach. aelite [30] and dAElite [31] have been proposed as improved next generation Æthereal NoCs. aelite inherits the guaranteed services model from Æthereal. To overcome the global synchronicity problem, aelite proposes use of asynchronous and mesochronous links as a possibility. Both aelite and dAElite are compact, efficient designs. A complete arity-5 aelite router with mesochronous links occupies 0.032 mm^2 and operates at more than 800 MHz in 90 nm CMOS technology. A 16 bit, 4 port dAElite router occupies 0.020 mm^2.

2.3. Label switching in NoCs

Label switching is used by technologies such as ATM [32] as a packet relaying technique. Individual packets carry route information in the form of labels. Routers along the path use the label to identify the next hop, forwarding information, traffic priority, Quality of Service guarantees and the next label to be assigned.

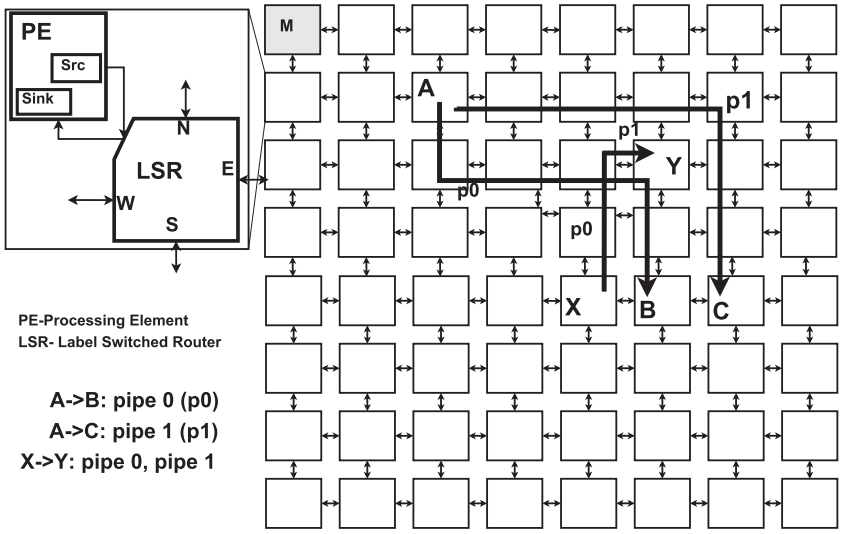


Fig. 1. A 64 Node, 8 × 8 2D LS-NoC with pipes between A and B (p0), A and C (p1), and X and Y (p0, p1) nodes.

Label switching inherently supports traffic engineering, as labels can be chosen based on desired next hop or required QoS services.

Use of Multi-Protocol Label Switching (MPLS) [33] for QoS [34] in NoCs and advantages of identifying communication channels using labels have been investigated in [12,35]. The work in [34] is a direct mapping of MPLS in the Internet to NoCs, modeled in Network Simulator 2 (NS2). The router and NoC designs are not optimized for a hardware implementation.

Implementation of label-based addressing in streaming applications have resulted in significant reduction in router area [35]. BlackBus [35] presents a simple data transfer scheme between communicating nodes and does not give special attention to delivering QoS.

2.4. Label switched NoC

Use of a centralized NoC Manager to identify resource-rich paths between communicating nodes is the unique feature of LS-NoC. In the proposed work, we describe a Label switched, QoS guaranteeing NoC that retains advantages of both packet switched and circuit switched networks. The key features and contributions of the work follow.

2.4.1. A label switching NoC providing QoS guarantees

The LS-NoC services QoS demands of streaming applications with the help of a centralized NoC Manager. LS-NoC guarantees deterministic path latencies, satisfies bandwidth requirements and delivers constant throughput. Delay and throughput guaranteed paths (*pipes*) are established between source and destinations along contention-free, bandwidth provisioned routes. Multiple pipes share the underlying physical link as long as their individual QoS requirements are met. Pipes are identified by *labels* unique to each source node. Labels need fewer bits compared to node identification numbers – potentially decreasing memory usage in routing tables.

2.4.2. NoC manager with traffic engineering capabilities

The NoC Manager utilizes flow identification algorithms to identify bandwidth rich, contention-free *pipes*. The LS-NoC Manager has complete visibility of the state of LS-NoC. Bandwidth requirements of the application are taken into account to provision routes between communicating nodes by the flow identification algorithm. Topology independent flow-based pipe establishment algorithm enables NoC Manager to support both regular NoCs in CMPs and customized NoCs in SoCs. The NoC Manager can be a software process running on a core or a separate hardware accelerator block. The Manager configures the routing tables in routers along the path of the pipe to complete the establishment of the pipe. Additionally, fault tolerance is achieved by the NoC Manager by considering link status during pipe establishment.

2.4.3. Design of a label switched router

The Label Switched (LS) router used in LS-NoC achieves single-cycle traversal delay during contentionless traffic, is multicast and broadcast capable. Source nodes in the LS-NoC can work asynchronously as cycle level scheduling is not required in the LS router. LS router supports multiple clock domain operation. LS-NoC enables circuit switching without requiring a globally synchronous clock and hence eases clock tree design and reduces global clock distribution power.

3. Label switched NoC

3.1. LS-NoC design

Fig. 1 shows an Example 8 × 8 LS-NoC in a 2-dimensional mesh topology. The LS-NoC Manager is placed in Node 0. Each of the blocks can contain identical processing elements or have custom IP blocks. Communication may be required between any number of blocks based on the application being executed in the processing elements.

In Fig. 1, pipes, p0 and p1, connect node A to B and C. Starting from node X, pipe p0, has been configured to Y. At an intermediate node, labels conflict between pipes from A to B and X to Y. Pipe p0's label is swapped to p1 in the last hop.

3.2. Label switched router design

The Label Switched (LS) router (Fig. 2) enables single-cycle flit traversal during contentionless traffic. Labels are sent through a dedicated set of links beside the data. Separating label from the data link reduces metadata management at the network interfaces at the ingress and egress of LS-NoC. Wires are relatively inexpensive in CMPs [36] – a 4 bit label incurs an overhead of 1.5% on a 256 bit wide data link. The accompanying label on the data bus is used to identify the intended outgoing port by the routing table.

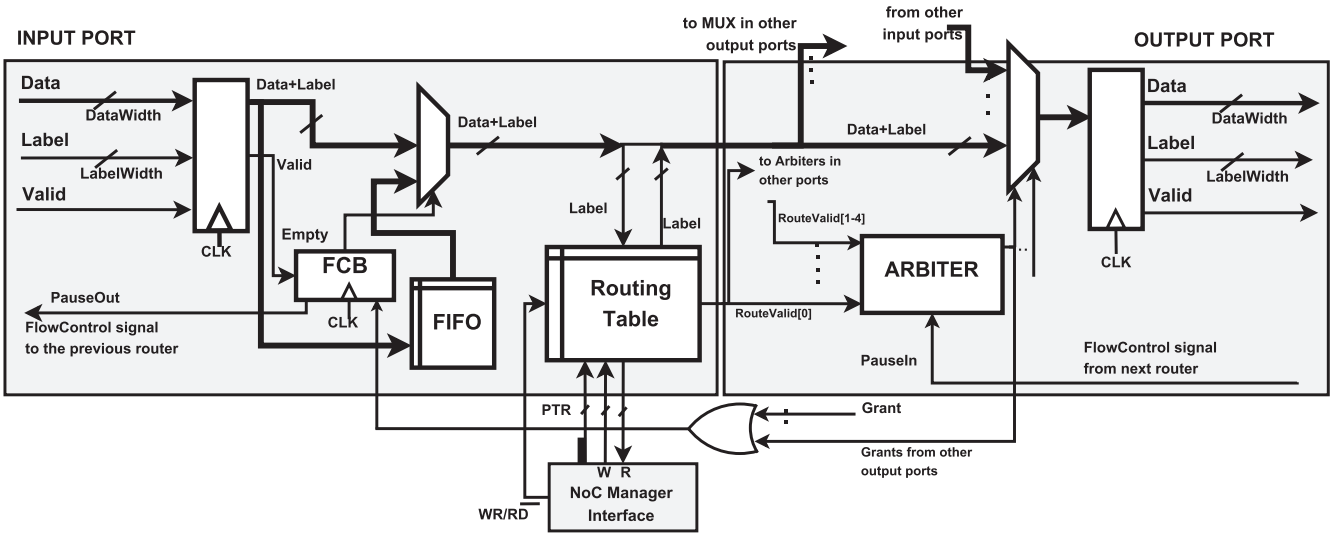


Fig. 2. Label switched router with single-cycle flit traversal. A valid signal identifies Data and Label as valid. PauseIn and PauseOut are flow control signals for downstream and upstream routers. Routing table has output port and label swap information. Arbiter receives input from all the input ports along with the flow control signal from the downstream router.

The combinational circuitry between a single input port to an output port in the label switched router is shown in Fig. 2. Incoming data flits are written into the FIFO if other flits are awaiting traversal or if arbiter does not grant access to the output port. The FIFO Control Block (FCB) handles FIFO pointer arithmetic and controls flow control signal of the corresponding input port.

The routing table in the LS-NoC router is indexed against established labels and has two fields, *Bit Mask* and *New Label* (Table 1). The *Bit Mask* contains as many bits as number of output ports in the router. A bit corresponding to an output port is set if the label to be routed is to exit the router from that output port. Multiple bits set in Bit Mask enable multicast and broadcast. The *New Label* field is maintained to enable label swapping (Section 3.4).

3.3. Pipes and labels

Persistent, throughput demanding connections between communicating nodes motivate the use of pipe-based communication in LS-NoC. A pipe is identified by the source, destination and a throughput guaranteed path between source and destination nodes.

A **pipe**($\mathscr{P}$) is a triplet $(\mathscr{S},\mathscr{K},\mathscr{R})$ where $\mathscr{S}$ and $\mathscr{K}$ denote source and sink nodes and $\mathscr{R}$ is a non-empty set of intermediate routers connecting $\mathscr{S}$ and $\mathscr{K}$. A source and a sink can have multiple pipes with varying set of intermediate routers, $\mathscr{R}$.

A **label** belonging to a source $\mathscr{S}$ uniquely identifies a communication pipe and the intended destination $\mathscr{K}$ – though the value of the label can change en route. Labels can be reused across sources. Each input port has a separate routing table up to 2^{lw} entries, where *lw* is the label width. Independent routing tables at input ports enable label reuse resulting in more efficient usage of label space. Label reuse by sources may give rise to label collision by pipes sharing a link. Label swapping reassigns unused labels to avert label collisions at input ports.

3.4. Label swapping

With increasing number of pipes in the LS-NoC, the probability of label collision increases. Label collision in a link results in routing table entry clash as shown in Fig. 3. Pipes entering North and South ports of Router 0 both have labels as 0. Both pipes are destined to leave the router through the East port and reach the West port of Router 1. There is no label conflict in Router 0 (contention exists at East port) as routing tables are private to each input port. Conflict occurs at the West port of Router 1 in Routing Table 2 through which both pipes need to be routed. Consider that the routing table entry for label 0 is already used and there are at least 2 routing table entries free for use. In such a situation, neither of the pipes having label 0 from Router 0 can pass through the West port of Router 1.

Label swapping reassigns labels to conflicting pipes using available label space at the next router. This allows complete utilization of the available label space. Fig. 3 illustrates label swapping for conflicting pipes. Routing Table 0 at the North port of Router 0 swaps conflicting label 0 to the available (in Router 1) label 1. Similarly, routing Table 1 at the South port of Router 0 swaps label 0 to 2 ensuring both pipes are setup to pass through West port of Router 1.

3.5. Flow based pipe identification

The LS-NoC Manager manages pipe establishment and tear down by taking into account the capacity required and capacity reserved in links. Flow identification algorithms [37,38] are used to identify available pipes between communicating entities. Ford-Fulkerson maxflow algorithm [38] is a classic solution for identifying a flow between two nodes in a graph. The flow algorithm has the complexity $\mathcal{O}(E.f)$ where E is total edges in the network graph and f is the number of flows to be identified. The Ford-Fulkerson algorithm was chosen for ease of implementation and its ability to converge in polynomial time.

Table 1
Routing table of a n port ($n = 5$) router with a lw bit ($lw = 4$) label indexed by labels used in the label switched NoC. Size of the routing table = $2^{lw} \times n \times lw$.

Input label	Direction bits	New label
0000	00001	0000
0001	00000	0001
0010	11111	0010
...	...	...
1111	00101	1111

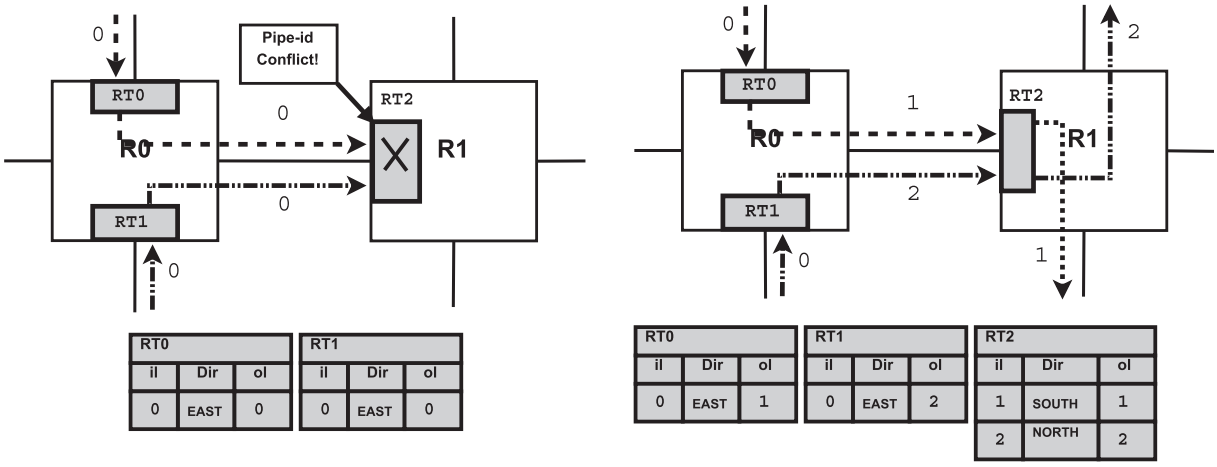


Fig. 3. Label conflict and label swapping.

The maximum capacity of a link is a measure of the peak bandwidth a link can support. Reserving required capacity ensures that adequate bandwidth is allocated for the pipe. Once a link has exhausted its capacity, no new pipes can be set up through it. Setting up an end-to-end flow after reserving requested capacity along the links between two communicating nodes ensures a QoS guaranteed route. Algorithm 1 is used by the NoC Manager to establish a QoS guaranteeing pipe in an LS-NoC.

One of the inputs to the algorithm is the flow graph of the NoC. A flow graph is a representation of communication between the nodes (n) in an NoC. The source nodes of the NoC flow graph are the processing engines that generate traffic and the router ports that forward traffic. Processing engines and router ports receiving data form the sink nodes in the flow graph. The connections between these communicating nodes are represented using edges. An edge, E_{ij}, from the flow graph is represented as:

$$E_{ij} = \left\{ n_i, n_j, u_{ij}, v_{ij}, L_{ij}^{used}, L_{ij}^{unused} \right\} \quad (1)$$

where edge E_{ij} connects nodes n_i to n_j. Nodes can be traffic sources, traffic sinks, router input or output ports. u_{ij} and v_{ij} are utilized and available flow capacities of the edge respectively. L_{ij}^{used} is the list of labels used in pipes through E_{ij}. L_{ij}^{unused} is the list of available labels to assign to future pipes. During the initialization stage, u_{ij} and L_{ij}^{used} are equal to null; v_{ij} holds the maximum capacity value E_{ij} can support and L_{ij}^{unused} is the list of all available labels for pipes through E_{ij}. Edges ending at the input ports of routers support a maximum capacity equal to the maximum pipes supported by the edge ($2^{lw} = 16$). These are the bottleneck edges. Edges not ending at input ports of routers are assigned infinite capacity and do not affect the final pipe route. A pipe of requested capacity, c, has to be established between the source and destination nodes, s and d.

Algorithm 1. Identify Pipe. P: Pipe Stack.

```
1:  Identify_Pipe
Require: Input Network:
    {E_ij} = {..., {n_i, n_j, u_ij, v_ij, L_ij^used, L_ij^unused}, ...},
    Source s, Destination d, Required Capacity: c
2:  Residual Graph, RG = {E_ji} = {..., {n_j, n_i, v_ij}, ...}
3:  EdgesCount: k ← 0
4:  if s = = d then
5:    push s onto {P}
6:    return true
7:  else
8:    for all edges starting from d do
9:      if v_dj > c then
10:       if call Identify_Pipe (RG, j, s, c)==true then
11:         push j onto {P}
12:         return true
13:       else
14:         pop from {P}
15:       end if
16:     end if
17:   end for
18: end if
19: return false
```

The first step in the algorithm is to build a *Residual Flow Graph* from the input flow graph. The residual flow graph contains the same number of edges as the flow graph – with directions reversed. All edges E_{ij} change to E_{ji} in the residual graph. The residual graph stores the available capacities of every link. The residual graph is used to identify a flow that satisfies requested capacity starting from the desired destination, d and traversing through the graph depth-first, till source, s, is found (Lines 8–17).

Starting from the destination node d, edges satisfying requested capacity are searched in the residual graph. If an edge satisfies the requested capacity, it is stored in the pipe stack ($\{P\}$). The next edge is searched from the terminating node of the previous edge (Line 10). If a node is reached from which no edge satisfies the requested capacity, the edge is popped out of the pipe stack (Line 14) and the search is backtracked. If the source has been reached through edges servicing requested capacity, the final edge is pushed into the pipe stack and flow has been identified (Lines 4–6).

After pipe has been identified, the pipe stack $\{P\}$ is used to update the used (L_{ij}^{used}) and available (L_{ij}^{unused}) flow capacities from the flow graph. During the configuration of routing tables, label swapping is performed in intermediate routers wherever necessary. An

Table 2
Simulation parameters used for functional verification of the label switched router design.

Network	64 node, 8 × 8, 2D mesh (Fig. 1)
Data bus width	256 bits
Label width	4 bits
Input buffer depth	8
Simulation time	10^8 cycles
Simulation framework	Icarus Verilog [39]

available label from the first router input node in $\{P\}$ is used. Along the route through $\{P\}$, L_{ij}^{used} of each edge is checked for conflicts. If a conflict occurs, an unused label from the pipe is used. The routing table data structure at a node, n_i, can be represented as:

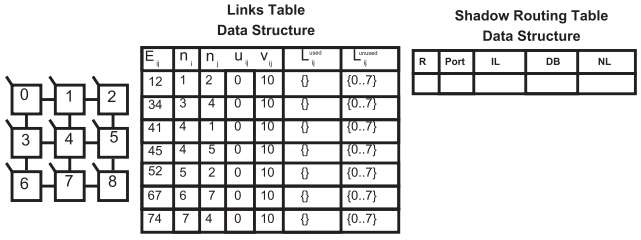
$$n_i, l_{old} \rightarrow n_j, l_{new} \tag{2}$$

where l_{old} is the label of the pipe in the edge ending at n_i and l_{new} is the label of the pipe in the edge E_{ij}. The procedure is repeated for every node along the pipe. This data structure is used by the NoC Manager to update routing tables along $\{P\}$.

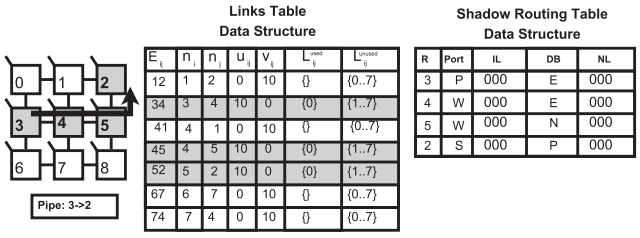
3.6. LS-NoC – Working

Fig. 4a–e illustrate a pipe establishment and label swapping example in LS-NoC. A shadow copy of individual routing tables can be stored at the NoC Manager's side, though this is not essential. The NoC Manager stores the number of pipes passing through the link. This information is used while building the network graph during pipe establishment.

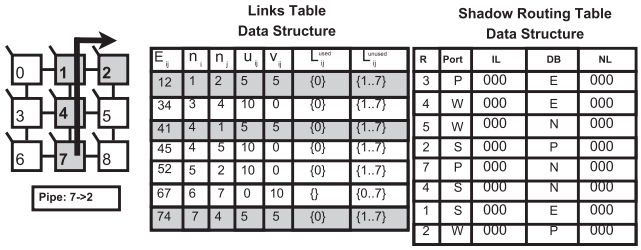
- Fig. 4a: NoC Manager contains the flow graph of the network. A few of the edges stored in a data structure are shown in this figure. Total labels available = 8. At the initialization stage all labels are free for use.
- Fig. 4b: A pipe ($3 \rightarrow 4 \rightarrow 5 \rightarrow 2$) is set up between nodes 3 and 2. The data structure is updated in rows associated with edges 34, 45 and 52. The pipe occupies the entire bandwidth of the link and uses label = 0.
- Fig. 4c: Another pipe is established between 7 and 2. The pipe is established in a non-intersecting manner w.r.t. to previous pipe. The pipe ($7 \rightarrow 4 \rightarrow 1 \rightarrow 2$) occupies 50% of the bandwidth available from the links.
- Fig. 4d: A pipe has to be established between nodes 6 and 4. The flow algorithm has identified the route $6 \rightarrow 7 \rightarrow 4$ for the pipe. Label 0 is the first available label at node 6. The label conflicts at the South port of node 4 – Label 0 has been utilized by pipe 0 of node 7 ($7 \rightarrow 2$).
- Fig. 4e: Label swapping enables pipe to be established with label 1.



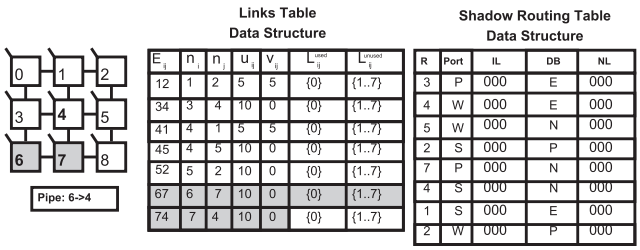
(a) Initial state of a few edges in NoC Manager.



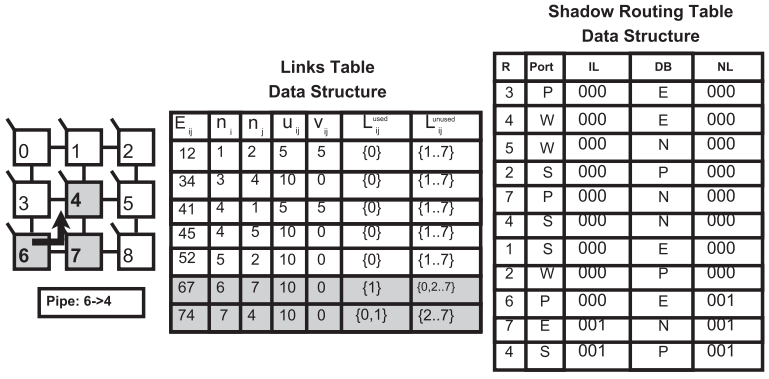
(b) LS-NoC state after pipe 3→2 has been established.



(c) LS-NoC state after pipe 7→2 has been established. The pipe has no contention with the previously established pipe.



(d) Label conflict at node 7's North port during 6→4 pipe establishment.



(e) Label swapping completed and label 1 assigned to pipe 6→4.

Fig. 4. Pipe establishment and label swapping example in a 3 × 3 LS-NoC.

3.7. Multiple clock domains and Fault Tolerance

The FIFO based router design in LS-NoC supports multiple clock domains with a little modification. The buffers at the output port in the router can be replaced by multi-clock or mesochronous buffers and can be connected to dual clock interfaces.

Fault tolerance is inherently built into LS-NoC because of the nature of pipe establishment process. At the onset of pipe establishment process, the updated graph of the NoC is constructed. During this stage, a failed link will have a capacity value of 0. The list of actions taken by the LS-NoC Manager after a faulty link has been recognized are listed here:

- On detection that a link is faulty, the LS-NoC Manager updates its capacity to 0 in the flow graph.
- Existing pipes through the link are invalidated. The pipes are re-identified and routing tables are updated.
- Flow graph is updated after pipes are configured.

3.8. Simulation and functional verification

Functional verification of the Verilog designs of the router and networks were done using Icarus Verilog Simulator [39]. Table 2 lists simulation parameters used in functional verification of the label switched router. Routing tables are populated in the initialization stage. The flow identification algorithm (Algorithm 1) is implemented in Perl. Routing tables in the NoC are configured based on the algorithm. The label switched router has been implemented and tested for the following networks:

- *Single router network*: A single router connected to four nodes (four sources and four sinks) was tested for 10^8 cycles for different directed traffic and random traffic permutations.
- *2D Mesh*: A 64 node, 64 router two dimensional mesh with traffic from each source to random destinations for 10^8 cycles was tested.

4. LS-NoC management

4.1. NoC manager and traffic engineering in LS-NoC

The NoC Manager identifies QoS guaranteed *pipes* between nodes and updates routing tables along the pipe. The NoC Manager can be implemented as a software thread running on one of the cores (in a CMP) or as a separate hardware accelerator (in an SoC). The ability of the Manager to establish pipes dynamically considering network traffic and availability of contention-free routes allows for guaranteed bandwidth and deterministic latency pipes.

The Manager monitors status of links in the NoC. The Manager works as a fault tolerant route set up system in the LS-NoC. Study of fault management strategies for LS-NoC using the Manager is not dealt with in this work. NoC Manager interfaces with routers and updates routing tables while new pipes are identified in the NoC. In SoCs where applications are fixed during operation, this will incur a one time set up latency at the initialization phase of the application.

The NoC Manager has complete visibility of current state of pipes in the NoC. Traffic engineering capabilities of LS-NoC enables flit forwarding through non-shortest, non-congested paths resulting in deterministic flit traversal latencies. Traffic engineering abilities of NoC Manager depend on pipe establishment algorithms (Section 3.5) and is independent of the network. This enables support for custom, specific application SoCs that contain non-homogeneous or adhoc NoCs. A centralized NoC Manager is adequate in most CMPs running streaming applications as there are a fixed number of communicating entities and the SoC is not a dynamically growing system. In a standard SoC, loss of scalability may not be a serious concern.

Table 3
NoC manager overhead.

	T_{comp}	T_{conf}	$T_{overhead}$
Single pipe	35157 cycles	17 cycles	35174 cycles (35.2 μs)

4.2. Overhead of NoC manager

Detailed analysis of the amount of time spent in identifying pipes and updating routing tables in LS-NoC is presented in this section. Overhead of the NoC Manager comprises of two components: computation and configuration. Computational overhead involves identifying a pipe using flow-based algorithm (Algorithm 1). Configuration overhead includes transmitting routing table configuration over the network and updating routing tables (Table 3).

4.2.1. Computational latency

The upper bound on pipe establishment time using the flow algorithm implemented in the NoC Manager is presented in this section. The LS router is equipped with an interface to read from/write into a single port routing table memory structure (Fig. 2). The NoC Manager process resides in the first node as shown in Fig. 1. The flow algorithm (Algorithm 1) has the complexity $\mathcal{O}(Ef)$. Total edges, E, in the graph representation of a 2D Mesh with degree d, grows as $2 \times d^2$. Total number of flows to be established depends on the applications and the number of process nodes. Computational overhead of establishing a single pipe in an 8×8 LS-NoC is presented in Table 3. The algorithm was executed on a Cortex A8 processor (ARM v7 architecture, operating frequency = 1 GHz). 35157 cycles were spent for identification of a single pipe. Identification of a single pipe involves building the residual graph from the flow graph, identifying a bandwidth satisfying pipe between source and destination and updating the flow graph of the NoC (Algorithm 1). As the number of steps in-

Table 4
(a) Synthesis parameters. (b) Synthesis results 2 router and mesh networks. Area of a router is 0.431 mm^2.

(a)	
Process	UMC 130 nm, high speed
Library	Faraday
Process	1.00
Temperature	25 °C
Voltage	1.2 V
Interconnect model	Worst case tree
Metal layers	8.2 thick layers
Router details	

(b)		
Ports	5	
Data width	256 bits	
Label width	4 bits	
	LS-NoC router	
	Buffers + routing table (storage elements)	Combinational logic
Synthesis results		
Area (mm^2)	0.077	0.354
Total area (mm^2)	0.431	
Router power	11.01 mW	32.07 mW
Total power	43.08 mW	
Max frequency	312.5 MHz	
Bandwidth/link	80 Gbps	

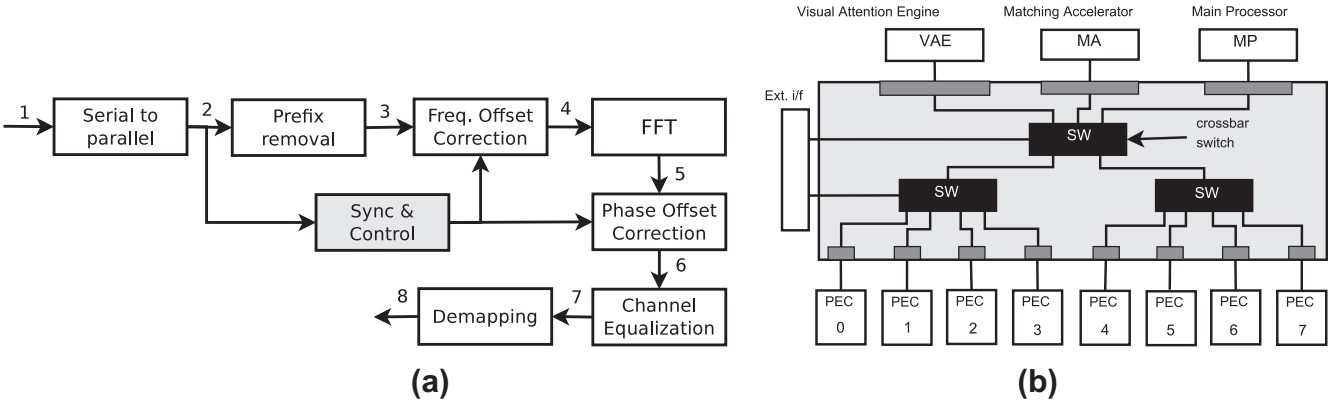


Fig. 5. (a) Process graph of a HiperLAN/2 baseband processing SoC [6] and (b) NoC of the object recognition processor [7].

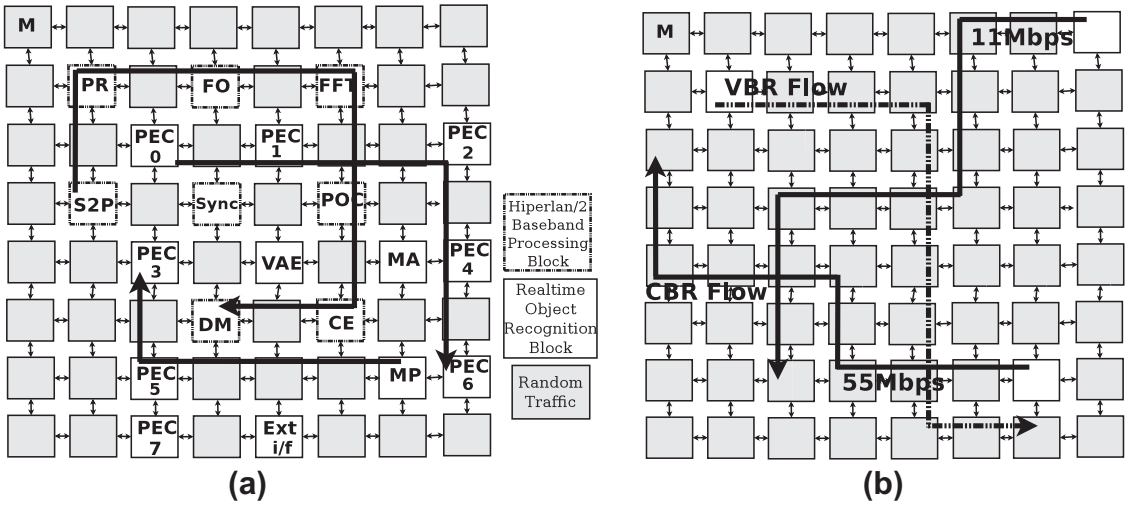


Fig. 6. (a) Process blocks of HiperLAN/2 baseband processing SoC and object recognition processor mapped onto a 8 × 8 LS-NoC. Pipe 1: PEC0 → PEC6, Pipe 2: MP → PEC3. (b) Flows set up for CBR & VBR traffic.

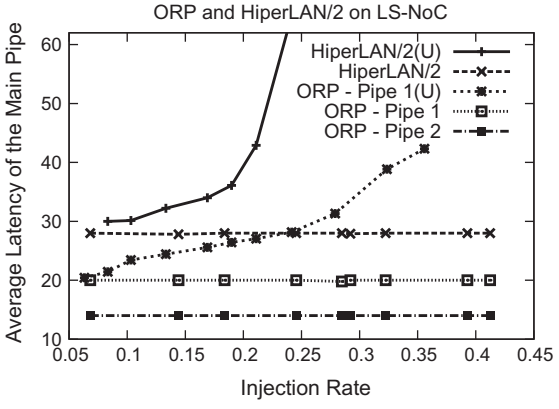


Fig. 7. Latency of HiperLAN/2 and ORP pipes in LS-NoC over varying injection rates of non-streaming application nodes. Latency of non-provisioned paths are titled (U).

volved in a pipe identification are the same, the time taken to identify p pipes is $p \times$ (time taken to identify one pipe).

4.2.2. Configuration latency

The worst case configuration overhead to update the routing table in the LS-NoC is 17 cycles. In the case of maximum pipes setup where all ports of all routers in the LS-NoC have to be updated by the NoC Manager, $T_{conf} = 5372$ cycles. Time to configure in the maximum pipes is derived as follows:

$$T_{conf} = T_{network} + T_{rt} \text{ cycles} \tag{3}$$

where $T_{network}$ is the network latency to transmit routes on the network and T_{rt} is the time to update routing tables in a router. In the worst case where all routers have to be updated, assuming a regular 2D Mesh,

$$T_{network} = (deg + 1) \times \left\{ \frac{deg \times (deg - 1)}{2} \right\} \text{ cycles} \tag{4}$$

$$T_{rt} = Size_{rt} \times deg^2 \text{ cycles} \tag{5}$$

Table 5
Pipes set up for HiperLAN/2 baseband processing SoC and object recognition processor SoC (Fig. 6a). PEC[0–7] → PEC[0–7]: every PEC communicates with every other PEC.

HiperLAN/2 baseband processing SoC
S2P → PR → FOC → FFT → POC → CE → DeMap,
S2P → Sync, Sync → POC, Sync → FOC
Object recognition processor
PEC[0–7] → PEC[0–7], MP → PEC[0–7], MP → VAE,
MP → Ext. I/f, MP → MA, VAE → PEC[0–7], MA → MP.

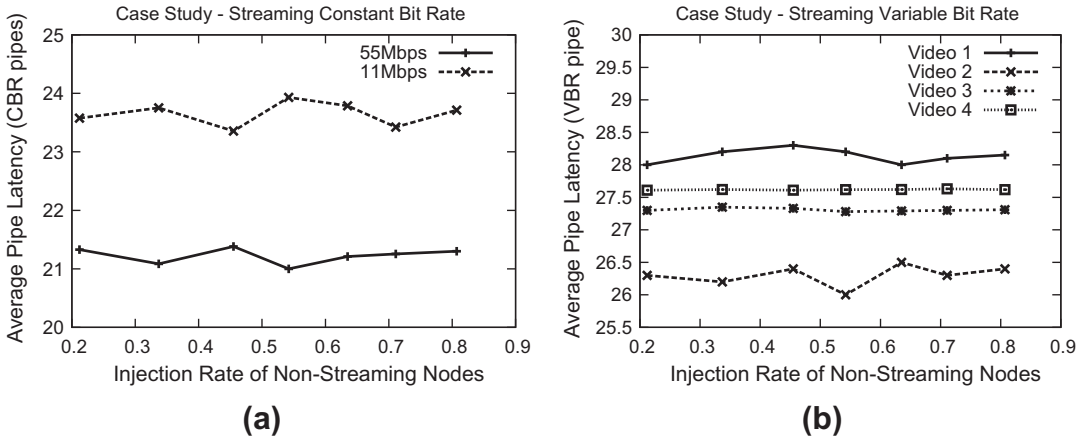


Fig. 8. (a) Latency of CBR traffic over various injection rates of non-streaming nodes in LS-NoC. (b) Latency of VBR traffic over various injection rates of non-streaming nodes in LS-NoC.

Table 6
Standard test videos used in experiments.

S.no.	Video	Frames per second	Frames simulated
1	Bridge close	10	200
2	Flower	10	250
3	Foreman	10	300
4	Hall	10	300

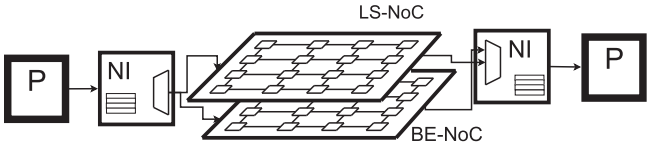


Fig. 9. LS-NoC being used alongside a best effort NoC.

where *deg* is the degree of the regular 2D Mesh, $Size_{rt}$ is the size of the routing table (number of writes required to fill the routing table). In the current work, *deg* = 8, $Size_{rt}$ = 80(16 per port × 5 ports), T_{conf} = 5372 cycles.

Given the polynomial completion time (complexity: $\mathcal{O}(E.f)$) of the pipe establishment algorithm, we envision that the time required for pipe establishment will not be prohibitive with respect to the application's lifetime. Consider the example of a video surveillance application that processes transmitted raw video stream in the video server. Such an application persists continuously for days together. The implemented pipe establishment algorithm spends less than half a second to establish a pipe and this delay is negligible considering the lifetime of the application. A centralized NoC Manager is adequate in CMPs that execute streaming applications. Such applications have a fixed number of communicating entities and the CMP is not a dynamically growing system.

5. Experiments and results

5.1. Synthesis results

Logic and combinational circuits blocks of LS Router were synthesized using Faraday's FSC0H_D 130 nm library, using high-performance and high-density generic core cells for UMC 0.13 μm eHS (FSG) process in the typical-NMOS, typical-PMOS case (Table 4a). Timing and area results from synthesis are tabulated in Table 4b. The router operates at 312.5 MHz in 130 nm technology. Taking into account effects of scaling, estimated frequency of operation of the router at 45 nm is above 1.2 GHz [40]. The link width of 256 bits was chosen to service peak bandwidth per link of 300 Gbits/s, comparable with GDDR5 bandwidth. Area and power of memory elements inside the LS Router (input buffers and routing tables) were estimated using UMC's Memory Compiler.

Synthesis of the functionally verified Verilog HDL design of the router was performed in Synopsys Design Compiler. Switching activity, timing and design constraints and the synthesized netlist are input to Cadence SOC Encounter. Area of the LS-NoC router is recorded after the place and route in SoC Encounter. Placed and routed netlist along with parasitics extracted file (SPEF) was used to obtain power of a router with no buffers. Area and power measurements of memory components for the buffers were done using UMC's Memory Compiler tool. The total power consumption is estimated to be 43.08 mW at 312.5 MHz, 1.2 V.

Area of a processing engine cluster from [7] was used to estimate the area as a case study to identify wire lengths in the mesh and feasibility of single-cycle operation. Area of a PEC was estimated as equal to 2.538 mm^2 and the length of a side is 1.593 mm. Intacte [41] was used to estimate the maximum frequency of operation of a 1.6 mm link in 130 nm. The 1.6 mm link can operate up to 1.5 GHz without pipelining. Single cycle packet transfer can be achieved when the entire NoC is operating at a frequency equal to the router's (312.5 MHz).

5.2. Case study – HiperLAN/2 baseband processing + object recognition processor SoC

HiperLAN/2 baseband processing SoC (Fig. 5a) [6] and Realtime objection recognition processor (Fig. 5b) [7] were mapped onto a 8 × 8 LS-NoC (Fig. 6a). Application pipes were setup as shown in Fig. 6a and Table 5. Hiperlan/2 blocks connect in a pipelined manner – the previous block's output serves as input to the next block (Fig. 5a). Serial-to-Parallel (S2P) block generates Hiperlan/2 application traffic to serve the Demapping block (DM) throughout the simulation. In the case of ORP SoC, the Main Processor (MP) orchestrates computational data between processor engine clusters (PEC), Visual Attention Engine and the Matching Accelerator. External interface is provided for off-chip multimedia data input/output. MP broadcasts periodically to all PECs while PECs set up bursty communications between themselves. Although the communication pipes are fixed over the experiments, the destinations for bursty and non-bursty traffic are chosen at random. Results for

Table 7
Evaluation of the proposed label switched router and NoC. CS: circuit switched, PS: packet switched.

Reference (router type)	Tech. (L), voltage	Area (A), (mm^2)	Router power (P), (mW)	Through-put (T), (per link, Gbps)	Energy efficiency (Tb/s/W)	FoM	Support async clock domains	Share link?
This work (LS) 8 × 8 Mesh	130 nm, 1.2 V	0.431	43.08	80	1.85	9.6	Yes	Yes
Nexus [12]	130 nm 1.2 V	1.75	–	48.75	–	–	Yes	Yes
SoCBUS [13]	180 nm	0.06	–	16	–	–	No	No
SDM [43]	1.2 V	0.135	1.790	0.64 (20 MHz)	0.3575	29.72	No	Yes
Æthereal (CS) [10]	130 nm, 1.2 V	0.26	–	16	–	–	No	Yes
aelite [30]	90 nm,	0.032	–	25.6	–	–	Yes	Yes
dAElite [31]	130 nm, 1.2 V	0.020	–	–	–	–	Yes	Yes
8 × 8 Mesh (CS) [5]	45 nm, 1.1 V	0.030 (Approx.)	21–74	11.78	0.159–0.560	188.46	No	No
Realtime ORP (PS) [7]	130 nm, 1.2 V	0.2 (Approx.)	46	12.8	0.278	29.8	Yes	Yes
HiperLAN/2 baseband SoC (CS) [6]	130 nm, 1.2 V	0.05	17.2 (Est.)	17.2	1.0	2.8	No	No

two sample pipes, PEC0 → PEC6 and MP → PEC3 are shown. Communication pipes were set up in the applications' case as shown in Table 5. Nodes marked 'Random Traffic' inject uniform random traffic into LS-NoC at various injection rates to mimic data from processes not belonging to either of the streaming applications in the CMP.

Fig. 7 presents pipe latency results from the 8 × 8 LS-NoC. The 8 × 8 LS-NoC services a 64 node CMP which is executing streaming applications. X-axis records injection rates of non-streaming application nodes. The LS-NoC was configured to support the 64 node CMP shown in Fig. 6a. LS-NoC establishes higher capacity pipes between communicating nodes of streaming applications. Resources are provisioned in pipes based on bandwidth requirements. A pipe might occupy a major portion or all of the available capacity of a link (Line 18–19 in Algorithm 1). The demand capacity of a provisioned pipe (C_{req}) may be tuned based on bandwidth requirements of the pipe. This ensures contention-free pipe set up and guaranteed bandwidth for the pipe. A non-provisioned pipe will share link resources equally with other pipes resulting in increased latencies as injection rate increases. Latency curves of the non-provisioned pipe (labeled 'U') clearly fail to guarantee QoS compared to provisioned pipes. Variation in injection rates of traffic generated by non-application nodes does not effect the latency in provisioned pipes. From the graph, average latency of flits traversing the provisioned pipe is almost constant over varying injection rates of other source nodes. The average latency of packets in the network does not change over injection rates due to reservation and provisioning of LS-NoC resources during the execution of the application. Aggregate bandwidth of 120 Gbits/s at maximum injection rate satisfies the communication requirements of both HiperLAN/2 and Object Recognition Processor applications.

5.3. Case study – video streaming applications

LS-NoC has been tested on both Constant Bit Rates (1.55 Mbps and 55 Mbps) and Variable Bit Rate traffic. Flows for CBR and VBR traffic were set up assuming worst case spatial separation between producer and consumer nodes (Fig. 6b). Results for CBR and VBR experiments are shown in Fig. 8. Videos used in H.264 standards evaluation were used for VBR experiments and are tabulated in Table 6. It is observed that latency of CBR and VBR traffic is unaffected by varying injection rates of non-video sources. All flows in the LS-NoC are provisioned such that CBR/VBR traffic experiences least contention at the routers. Throughput requirements for CBR/VBR traffic are met and the pipes display deterministic latency in the LS-NoC.

5.4. Discussion

5.4.1. LS-NoC application

For streaming applications, results in Figs. 7 and 8 show that the LS-NoC displays predictable latency and guarantees throughput. LS-NoC is suitable for applications whose communication patterns do not vary during application execution and require guaranteed throughput.

In CMPs and complex SoCs, LS-NoC can be used as a separate NoC to service applications requiring hard QoS guarantees. Network interfaces at ingress of the NoC can be configured to identify traffic belonging to QoS classes. Based on the type of the traffic being injected into the communication medium, either the conventional best effort NoC or the LS-NoC can be chosen. Multiple NoCs to service individual classes of data have been present in commercially available multi-core chips [42]. The concept is illustrated in Fig. 9.

5.4.2. LS-NoC evaluation

Table 7 presents a comparative illustration of the proposed LS-NoC with a few proposed QoS NoCs. A link width of 256 bits was chosen to service a peak bandwidth comparable with the GDDR5 bandwidth. The high link-width results in a throughput larger than other designs in the LS-NoC router. Each port in the LS-NoC contains 8 input buffers each. Every buffer is 256 bit wide contributing to high area and power. Router area of designs in [5,7] were estimated from total chip area. Custom design and low buffer usage in these routers has brought down the area significantly. Nexus [12] implements a 16 port, 36 bit asynchronous crossbar resulting in high area consumption.

Voltage scaling enables Intel's NoC [5] to operate in the 21–74 mW range. Area and power consumption of the ORP router [7] are similar to the LS Router owing to similar buffer area. The 7 port router has 32 bit input buffers with up to 10 buffers per input port. The low power and area values of the HiperLAN/2 SoC router is due to the absence of input buffers and arbitration unit in the router. LS-NoC has maximum throughput owing to high link width. However, LS-NoC shows high energy efficiency due to high throughput at a nominal frequency of 312.5 MHz in 130 nm technology. Results show that LS-NoC is the best design in terms of bits transmitted per Watt consumed.

The normalized Figure of Merit ((Area × Power)/Throughput) is a technology independent comparison parameter. The Figure of Merit (FoM) of 180 nm and 130 nm designs are scaled to 45 nm by multiplying by the ratio of cubes of channel lengths as shown in Eq. 5.4.2.

$$\text{FoM} = \frac{\text{Area} \times \text{Power}}{\text{Throughput}} \times \frac{L_{45}^3}{L_{Tech}^3}$$

L_{Tech} is the channel length of the corresponding Technology. With constant voltage and frequency, power varies linearly with technology owing to effects of capacitance. Length and breadth vary linearly to contribute to squared relation with respect to technology [44]. Hence ratio of cubed channel lengths normalize the FoM. FoM is a measure of resources spent per bit transmitted (hence, lower the better). Buffer-less low area design of the HiperLAN/2 router contributes to the low FoM number for this design. Such a buffer-less circuit switched router cannot accommodate multiple connections through a physical link. Resource reservation by a single circuit for the lifetime of the application may result in inefficient network utilization in such networks. LS-NoC fares fairly well owing to high throughput – though the area cost is high (Estimated area of the LS-NoC router is 0.054 mm^2 after scaling quadratically to 45 nm technology). The high power consumption of 74 mW in 45 nm contributes to the maximum FoM number in Intel's circuit switched router. Globally synchronous designs demand power hungry single clock distribution over the chip. LS-NoC allows link sharing by pipes without bandwidth compromise. Physical link sharing by multiple pipes results in higher network utilization compared to a purely circuit switched NoC. From tabulated results, the LS router has high energy efficiency and provides hard QoS guarantees at a reasonable power budget.

6. Conclusion

Streaming applications have deterministic communication patterns due to pipelined nature of operation. Traffic engineering in LS-NoC guarantees QoS and delivers constant throughput in such applications.

The proposed LS-NoC services QoS demands of streaming applications using a traffic-engineering capable NoC Manager. The centrally-managed, bandwidth-provisioned NoC Manager utilizes flow identification algorithms to identify contention-free, bandwidth-provisioned paths. Network visibility enables NoC Manager to configure bounded latency pipes in homogeneous and heterogeneous networks alike. Flow identification algorithm takes into account bandwidth available in individual links, to establish QoS guaranteed pipes. The algorithm allows sharing of physical links between pipes without compromising QoS guarantees.

The Label Switched (LS) router used in LS-NoC has a single-cycle traversal delay for contentionless traffic. The router is multicast and broadcast capable. Bi-synchronous or mesochronous input buffers in LS router enable multiple clock domain operation, without globally synchronous clocks. A 5 port, 256 bit data bus, 4 bit label, 1 bit flow control, 8 buffers per input port individual router occupies an area of 0.431 mm^2 in 130 nm Faraday library in the typical corner and operates at 312.5 MHz. LS-NoC is estimated to consume 43.08 mW. LS-NoC has been evaluated over example streaming applications, CBR and VBR traffic, and QoS guarantees have been demonstrated. Overhead of setting up a pipe by a software LS-NoC Manager is 35174 cycles (up to 35.2 μs). A comparison of QoS based NoCs has been presented and the proposed work has been evaluated. A technology independent figure of merit: $\frac{\text{Area}\times\text{Power}}{\text{Throughput}} \times \frac{L_{45}^3}{L_{Tech}^3}$ has been used by the current work. Owing to the high throughput achieved, the LS-NoC router has a competitive FoM among compared designs.

We envision the use of LS-NoC in general purpose CMPs where applications demand deterministic latencies and hard bandwidth requirements. LS-NoC can be used as a separate layer, catering to applications requiring hard QoS guarantees.

References

[1] W. Dally, B. Towles, Route packets, not wires: on-chip interconnection networks, in: Proceedings of Design Automation Conference, 2001, pp. 684–689.

[2] A. Jantsch, H. Tenhunen (Eds.), Networks on Chip, Kluwer Academic Publishers., Hingham, MA, USA, 2003.

[3] L. Benini, G. Micheli (Eds.), Networks on Chips: Technology and Tools, Morgan Kaufmann, CA, USA., 2006.

[4] S. Bell, Tile64 – processor: a 64-core soc with mesh interconnect, in: Solid-State Circuits Conference Digest of Technical Papers (ISSCC), IEEE International, 2008, pp. 88–89.

[5] M. Anders, H. Kaul, S. Hsu, A. Agarwal, S. Mathew, F. Sheikh, R. Krishnamurthy, S. Borkar, A 4.1 tb/s bisection-bandwidth 560 gb/s/w streaming circuit-switched 8 × 8 mesh network-on-chip in 45 nm cmos, in: Solid-State Circuits Conference Digest of Technical Papers (ISSCC), IEEE International, 2010, pp. 110–111.

[6] P.T. Wolkotte, G.J.M. Smit, G.K. Rauwerda, L.T. Smit, An energy efficient reconfigurable circuit switched network-on-chip, in: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS05) – 12th Reconfigurable Architecture Workshop (RAW 2005), p. 155a. ISBN: 0-7695.

[7] K. Kim, J.-Y. Kim, S. Lee, M. Kim, H.-J. Yoo, A 76.8 gb/s 46 mw low-latency network-on-chip for real-time object recognition processor, in: Solid-State Circuits Conference, 2008 (A-SSCC '08), IEEE Asian, 2008, pp. 189–192.

[8] T. Wiegand, G. Sullivan, G. Bjontegaard, A. Luthra, Overview of the h.264/avc video coding standard, IEEE Transactions on Circuits and Systems for Video Technology 13 (2003) 560–576.

[9] T.C. Xu, A.W. Yin, P. Liljeberg, H. Tenhunen, A study of 3d network-on-chip design for data parallel h.264 coding, Microprocessors and Microsystems 35 (2011) 603–612.

[10] E. Rijpkema, K. Goossens, A. Radulescu, J. Dielissen, J. van Meerbergen, P. Wielage, E. Waterlander, Trade-offs in the design of a router with both guaranteed and best-effort services for networks on chip, IEE Proceedings Computers and Digital Techniques 150 (2003) 294–302.

[11] J. Liang, A. Laffely, S. Srinivasan, R. Tessier, An architecture and compiler for scalable on-chip communication, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 12 (2004) 711–726.

[12] A. Lines, Asynchronous interconnect for synchronous soc design, IEEE Micro 24 (2004) 32–41.

[13] D. Wiklund, D. Liu, Socbus: Switched network on chip for hard real time embedded systems, in: Proceedings of the 17th International Symposium on Parallel and Distributed Processing, IPDPS '03, IEEE Computer Society, Washington, DC, USA, 2003. p. 78.1.

[14] E. Beigne, F. Clermidy, P. Vivet, A. Clouard, M. Renaudin, An asynchronous noc architecture providing low latency service and its multi-level design framework, in: Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems, 2005 (ASYNC 2005), 2005, pp. 54–63.

[15] R. Marculescu, U. Ogras, L.-S. Peh, N. Jerger, Y. Hoskote, Outstanding research problems in noc design: system, microarchitecture, and circuit perspectives, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 28 (2009) 3–21.

[16] E. Bolotin, I. Cidon, R. Ginosar, A. Kolodny, Qnoc: Qos architecture and design process for network on chip, Journal of Systems Architecture 50 (2004) 105–128.

[17] D. Lattard, E. Beigne, F. Clermidy, Y. Durand, R. Lemaire, P. Vivet, F. Berens, A reconfigurable baseband platform based on an asynchronous network-on-chip, IEEE Journal of Solid-State Circuits 43 (2008) 223–235.

[18] F. Karim, A. Nguyen, S. Dey, An interconnect architecture for networking systems on chips, IEEE Micro 22 (2002) 36–45.

[19] E. Salminen, T. Kangas, T.D. Hämäläinen, J. Riihimäki, V. Lahtinen, K. Kuusilinna, Hibi communication network for system-on-chip, Journal of VLSI Signal Processing Systems 43 (2006) 185–205.

[20] X. Wu, Y. Wu, L. Wang, X. Yang, Qos router with both soft and hard guarantee for network-on-chip, in: NORCHIP, 2009, pp. 1–6.

[21] Y. Salah, R. Tourki, Design and fpga implementation of a qos router for networks-on-chip, in: 3rd International Conference on Next Generation Networks and Services (NGNS), 2011, pp. 84–89.

[22] K.-C. Chang, J.-S. Shen, T.-F. Chen, Evaluation and design trade-offs between circuit-switched and packet-switched nocs for application-specific socs, in: Proceedings of the 43rd annual Design Automation Conference, DAC '06, ACM, New York, NY, USA, 2006, pp. 143–148.

[23] T. Richardson, C. Nicopoulos, D. Park, V. Narayanan, Y. Xie, C. Das, V. Degalahal, A hybrid soc interconnect with dynamic tdma-based transaction-less buses and on-chip networks, in: 19th International Conference on VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design, 2006, 8 pp.

[24] J. Hu, Y. Deng, R. Marculescu, System-level point-to-point communication synthesis using floorplanning information, in: Proc. ASP-DAC, 2002, pp. 573–579.
[25] D. Castells-Rufas, J. Joven, J. Carrabina, A validation and performance evaluation tool for protonoc, in: International Symposium on System-on-Chip, 2006, pp. 1–4.
[26] T. Bjerregaard, J. Sparso, A router architecture for connection-oriented service guarantees in the mango clockless network-on-chip, in: Proceedings of Design, Automation and Test in Europe, vol. 2, 2005, pp. 1226–1231.
[27] J. Ouyang, Y. Xie, Loft: a high performance network-on-chip providing quality-of-service support, in: Proceedings of the 2010 43rd Annual IEEE/ACM International Symposium on Microarchitecture, MICRO '43, IEEE Computer Society, Washington, DC, USA, 2010, pp. 409–420.
[28] K. Goossens, J. Dielissen, A. Radulescu, æthereal network on chip: concepts, architectures, and implementations, IEEE Design Test of Computers 22 (2005) 414–421.
[29] E. Rijpkema, K. Goossens, P. Wielage, A router architecture for networks on silicon, in: Proceedings of Progress 2001, 2nd Workshop on Embedded Systems, 2001, pp. 181–188.
[30] A. Hansson, M. Subburaman, K. Goossens, Aelite: a flit-synchronous network on chip with composable and predictable services, in: Proceedings of the Design, Automation & Test in Europe Conference and Exhibition, IEEE Computer Society Press, Los Alamitos, 2009, pp. 250–255.
[31] R. Stefan, A. Molnos, K. Goossens, daelite: a tdm noc supporting qos, multicast, and fast connection set-up, IEEE Transactions on Computers 99 (2012).
[32] A.E. Joel, Asynchronous Transfer Mode Switching, IEEE, 1993.
[33] E. Rosen, A. Viswanathan, R. Callon, Multiprotocol Label Switching Architecture, RFC 3031 (2001).
[34] M. Kim, D. Kim, G. Sobelman, Network-on-chip quality-of-service through multiprotocol label switching, in: Proceedings of IEEE International Symposium on Circuits and Systems, 2006 (ISCAS 2006), 2006, p. 1843.
[35] M. Koibuchi, K. Anjo, Y. Yamada, A. Jouraku, H. Amano, A simple data transfer technique using local address for networks-on-chips, IEEE Transactions on Parallel Distribution Systems 17 (2006) 1425–1437.
[36] Tilera Corporation, TILE-Gx 3000 Series Overview, 2011.
[37] N. Megiddo, Optimal flows in networks with sources and sinks, Mathematical Programming 7 (1974) 97–107.
[38] L.R. Ford, D.R. Fulkerson, A simple algorithm for finding maximal network flows and an application to the hitchcock problem, Canadian Journal of Mathematics 9 (1957) 210–218.
[39] Icarus iverilog, 2011. <http://iverilog.wikia.com/>.
[40] ITRS, International Technology Roadmap for Semiconductors, 2011.
[41] R. Nagpal, M. Arvind, Y.N. Srikanth, B. Amrutur, Intacte: tool for interconnect modelling, in: Proc. of 2007 Intl Conf. on Compilers, Architecture and Synthesis for Embedded Systems (CASES 2007), 2007, pp. 238–247.
[42] TileraGX, Tile-GX Processor Family, 2011.
[43] A. Leroy, D. Milojevic, D. Verkest, F. Robert, F. Catthoor, Concepts and implementation of spatial division multiplexing for guaranteed throughput in networks-on-chip, IEEE Transactions on Computers 57 (2008) 1182–1195.
[44] N.H.E. Weste, D. Harris, CMOS VLSI Design – A Circuit and Systems Perspective, Addison Wesley 2011, p. 256.

Bharadwaj Amrutur (M'08) received the B.Tech. degree in Computer Science and Engineering from the Indian Institute of Technology, Bombay, India, in 1990 and the M.S. and Ph.D. degrees in Electrical Engineering from Stanford University, Palo Alto, CA, in 1994 and 1999, respectively. He has worked at Bell Labs, Agilent Labs and Greenfield Networks. He is currently an Associate Professor in the Department of Electrical Communication Engineering, Indian Institute of Science, Bangalore, India, where he is working in the areas of VLSI Circuits and Systems.

Basavaraj Talwar received the B.Tech degree in Computer Engineering from National Institute of Technology Karnataka, India in 2003 and M.Tech in Networking & Internet Engineering from SJCE, Mysore, India in 2005. He is currently pursuing graduate studies in the Electrical Communication Engineering Department of Indian Institute of Science, Bangalore, India. His research interests include power and performance effects of interconnection networks in System-on-Chips and Chip-Multiprocessors.