

# Garnet: A Detailed Interconnect Model Inside a Full-System Simulation Framework

Niket Agarwal, Li-Shiuan Peh and Niraj K. Jha

Department of Electrical Engineering

Princeton University

{niketa, peh, jha}@princeton.edu

## I. INTRODUCTION

*Garnet* is an interconnection network model inside a full-system simulation framework (*GEMS* [3]). It consists of a “fixed pipeline” model as well as an approximate model which is called the “flexible pipeline” model. The fixed pipeline model is intended for low-level interconnection network evaluations and models the detailed features of a state-of-the-art network. Researchers interested in investigating different network microarchitectures can readily modify the modeled microarchitecture and pipeline. Also, for system level evaluations that are not concerned with the detailed network characteristics, the fixed pipeline model provides an accurate network model and should be used. For evaluations that wish to easily tune some standard network parameters like router pipeline depth, buffer sizes, virtual channels, etc. the “flexible pipeline” model provides a neat abstraction that can be used. It models the interconnect network to a certain amount of detail while providing easy tuning of network parameters.

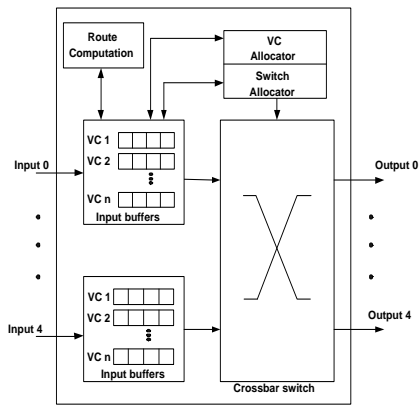
## II. FIXED 5-STAGE PIPELINE MODEL

### A. Router Design

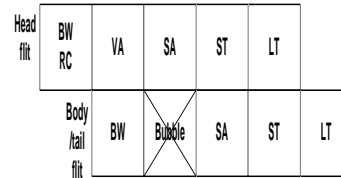
Figure 1(a) shows the microarchitecture of the state-of-art virtual channel (VC) router that is modeled. The router can have any number of input and output ports depending on the topology and configuration. The major components which constitute a router, are the input buffers, route computation logic, VC allocator, switch allocator and crossbar switch. Figure 1(b) shows the router pipeline that we model. Since on-chip designs need to adhere to tight budgets and low router footprints, we model flit-level<sup>1</sup> buffering and credit-based VC flow control at every router. Every VC has its own private buffer and its size can be specified at runtime. The routing is table based and deterministic.

A head flit on arriving at an input port, first gets decoded and gets buffered according to its input VC in the buffer write (BW) pipeline stage. Every VC has its own private buffer. In the same cycle, a request is sent to the route computation unit (RC) simultaneously, and the output port for this packet is calculated.

<sup>1</sup>A flit is the smallest unit of flow control. A packet consists of a head flit, followed by body flits, and ends with a tail flit.



(a) Router microarchitecture



(b) Router pipeline

Fig. 1. Classic 5-stage pipeline router

The header then arbitrates for a VC corresponding to its output port in the VC allocation (VA) stage. Upon successful allocation of an output VC, it proceeds to the switch allocation (SA) stage where it arbitrates for the switch input and output ports. On winning the switch, the flit moves to the switch traversal (ST) stage, where it traverses the crossbar. This is followed by link traversal (LT) to travel to the next node. Body and tail flits follow a similar pipeline except that they do not go through RC and VA stages, instead inheriting the VC allocated by the head flit. The tail flit on leaving the router, deallocates the VC reserved by the packet.

**Router microarchitectural components:** Keeping in mind on-chip area and energy considerations, single-ported buffers and a single shared port into the crossbar from each input were designed. Separable VC and switch allocators as proposed in [4] were modeled. This was done because these designs are fast and of low complexity, while still providing reasonable throughput, making them suitable for the high clock frequencies and tight area budgets of on-chip networks. The individual allocators are round-robin in nature.

### B. Features supported

1. **Configuration Parameters:** The following configuration parameters can be specified :

- **flit size** - number of flits per packet
- **number of virtual channels** - this specifies the number of VCs per virtual network
- **buffer size** - this specifies the number of flit size buffers per VC

2. **Point-to-point ordering:** Some coherence protocols require some of the virtual networks to support point-to-point ordering. This implies that if two messages are sent from a particular source node to the same destination node one after the other, the order in which they are received at the destination has to be the same in which they were sent. To provide this support, there are some features that the VC allocator and

the switch allocator should support. In the VC allocation phase, if there are contending requests from VCs from the same virtual network to the same output port for an output VC, then requests for the packet which arrived at the router first has to be serviced first. During switch allocation, if there are contending requests for the same output port from multiple VCs from the same input port (and belonging to the same ordered virtual network) , then the request for the packet that arrived at the router first will be serviced before others. This and deterministic routing guarantees that packets that have the same route can never get re-ordered in the network.

**3. Treating of multi-cast messages:** The network modeled does not have hardware multicast support within the network. Whenever the protocol sends a multicast message, it gets broken up into multiple unicast messages at the NI. Each unicast message is now treated separately in a similar fashion to any other message in the network.

**4. Modeling variable link bandwidth:** The flit size specifies the link bandwidth as - number of bytes per cycle per network link. Links with lower bandwidth than this (for instance some off-chip links) are modeled by specifying a longer latency across them.

**5. Deadlock avoidance:** Since the network modeled supports finite buffering and finite VCs, if there is a cyclic dependency between resources, a deadlock might occur in the system. A deadlock is avoided by not allowing such cyclic dependencies to occur. In our network, this is done by having support for dimension ordered routing in the network. The FILE\_SPECIFIED topology specification format has an entry, “link\_weight” for every link that can be specified. During the topology specification phase, links in one dimension can be given lower weights than links in another dimension. The original GEMS implementation populates the routing tables by calculating a minimal path between nodes. It allows for more than one minimal path to be added into the routing tables. During the route computation (RC) phase inside the routers, these weights are looked up and compared for the choice of more that one minimal path. The output link with the minimal weight is selected. Thus, to provide X-Y routing, in which a packet needs to traverse all the X direction hops first and then the Y direction hops, the X direction links are given a lower weight.

Figure 2 shows the latency of the fixed pipeline model as a function of increasing network traffic. The traffic injected was uniformly random in nature. The injection rate at each input node is calculated based on the fraction of the bisection bandwidth that gets saturated by the injection of traffic at that rate. Calculation were done as illustrated in [1]. Under very low network load, when most routers are free and there is less contention, the packet latencies are very less. However as load increases and contention for network resources becomes higher, a corresponding increase in network latency is observed. After a certain load (70% of network capacity), the network just saturates. These results are in accordance with previously published results [2], for the same network model under similar network traffic.

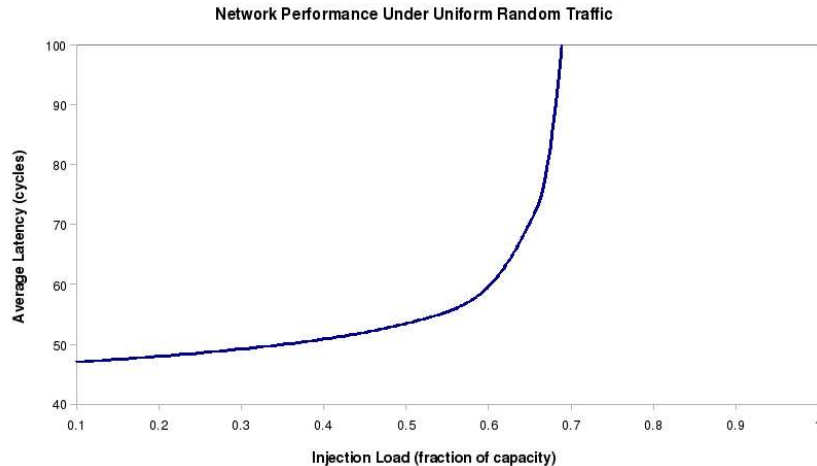


Fig. 2. Fixed pipeline model performance

### III. FLEXIBLE MODEL

#### A. Router Design

The “flexible-pipeline” model is intended to provide a reasonable abstraction of all interconnection network models, while allowing the router pipeline to be flexibly adjusted. A router pipeline might range from a single cycle to several cycles. The basic motivation for coming up with this model was the fact that not all interconnect designs assume the same router pipeline implementation. The “flexible-pipeline” model implements an output queued router that adds a variable number of cycles after a flit reaches an output of a router. This fakes the router pipeline delay. Output queued routers have the ideal router microarchitecture in that they provide 100% throughput by ideal utilization of buffers and links. It is not practical though to build output queued routers given the on-chip power and area budgets. We chose output queued routers for our flexible model because it provides an easy way of faking any pipeline delay by just adding variable latency after a flit is queued in an output queue.

A head flit on arriving at a router, moves to a destined output port and output VC which is predecided. The output port and VC are arbitrated for one hop in advance. This is because on arrival at the router, the flit has to be guaranteed a buffer. The flit gets buffered at the output queue for the number of pipeline stages specified. In the meanwhile, the head flit sends a VC arbitration request to the downstream router. The downstream router calculates the output port for the flit and looks for a free VC. On successfully getting a free VC, the router signals the upstream router that VC arbitration for the flit was done. The head flit now arbitrates for the output physical link and moves out of the router. Body and tail flits follow the same course except that they do not arbitrate for a VC. Tail flits also release the VC after they exit the router. A flit prior to its departure from a router, checks for free buffers in the downstream router. Thus finite buffering

is also modeled. This implementation models link contention, VC allocation, router pipeline delay and finite buffering. What it idealizes is the internal switch.

### B. Features Supported

1. **Configuration Parameters:** The following configuration parameters can be specified :

- **pipeline stages** - the number of pipeline stages in a router
- **flit size** - this directly controls the link bandwidth in the system
- **number of virtual channels** - this specifies the number of VCs per virtual network
- **buffer size** - this specifies the number of flit size buffers per VC

2. **Features same as detailed model:** Some features are modeled in the same way as that of detailed model - multi-cast messages, variable link bandwidth and deadlock avoidance.

3. **Point-to-point ordering:** For virtual networks that require point-to-point ordering, only 1 virtual channel is allocated. Thus, the messages from same source to destination travel one after the other and in a total order.

Figure 2 shows the latency of the fixed pipeline model as a function of increasing network traffic. The traffic injected was uniformly random in nature. The injection rate was calculated in a similar manner as in the experiments with the fixed pipeline model. A network with an output queued router model saturates when the injected load is very close to 100% of network capacity. This is because output queued routers provide close to 100% throughput and utilization. However, the flexible pipeline implementation models finite buffering and hence saturates when the injected traffic is around 90% of capacity. This is expected of this kind of a model. Since the flexible model is output queued in nature, it provides better throughput than the fixed pipeline model.

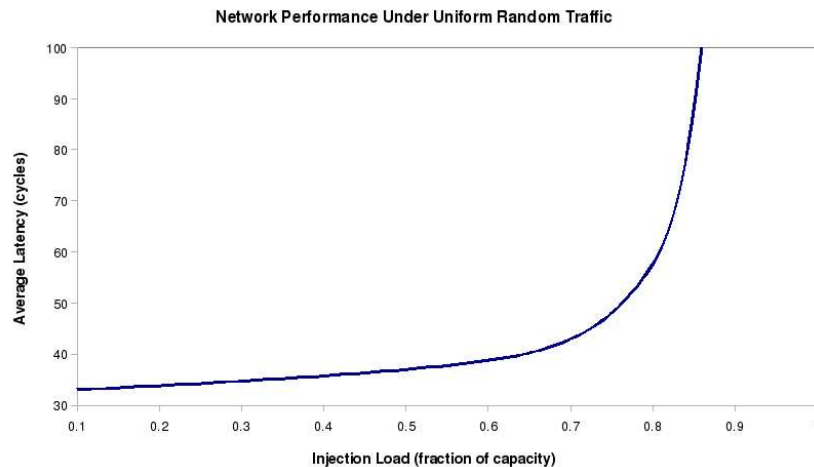


Fig. 3. Flexible pipeline model performance

#### IV. ORION POWER MODELS

The fixed pipeline model also has the *Orion* power models [5] incorporated. Various performance counters that are required for power calculations are recorded during the simulation. In the end, they are fed to Orion for reporting the interconnection network power. Both dynamic and leakage power are reported.

#### V. CONCLUSION

Garnet provides a platform for detailed interconnection network experiments and also enables more accurate full-system evaluations. More information on Garnet can be found at <http://www.princeton.edu/~niketa/garnet.html>

#### REFERENCES

- [1] W. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, CA: Morgan Kaufmann Pub., 2003.
- [2] A. Kumar, L.-S. Peh, P. Kundu, and N. K. Jha, "Express virtual channels: Toward the ideal interconnection fabric," in *Proceedings of ISCA-34*, 2007.
- [3] M. Martin, D. Sorin, B. Beckmann, M. Marty, M. Xu, A. Almadeen, K. Moore, M. Hill, and D. Wood, "Multifacet's general execution-driven multiprocessor simulator (gems) toolset," in *Computer Architecture News*, 2005.
- [4] L.-S. Peh and W. J. Dally, "A delay model and speculative architecture for pipelined routers," January 2001, pp. 255–266.
- [5] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: a power-performance simulator for interconnection networks," in *Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture*, 2002, pp. 294–305.