A Generalisation of Router Chip Design

Pauline Catriona Haddow
The Norwegian University for Science and Technology
Department of Computer Systems
7030 Trondheim, Norway
pauline@idt.ntnu.no

Abstract

To improve performance in parallel computers, the challenge is to reduce the communication bottleneck by decreasing latency and increasing throughput. There is currently much research into finding solutions to the bottleneck problem where the focus is on increasing throughput. However, many of these proposed solutions lead to an increase in complexity at the network level. This gives rise not only to an increase in latency but also an increase in costs. Traditional performance models have not, in general, focused on the cost issue. Most performance models are developed for a particular or restricted set of network designs and performance criteria and therefore can be said to be static in nature. New design ideas will often result in the development of a new model. Therefore, to create a detailed model applicable to both existing and future networks a more flexible model structure is required. This paper presents a model based on a generic router for cost versus performance analysis at the network level for a range of router designs.

1 Introduction

In a message-passing multicomputer each node contains an autonomous processor and local memory that is physically separate and logically private from the memories of the other nodes. Interprocessor communication occurs by routing messages explicitly through a message-passing network. An interface is required to control the flow of information between the processor and the interconnection network. The complexity of this interface depends on the quality of service offered by the computer system, the topology of the interconnection network and the division of work between the interface and the compute processor itself. This interface is termed a router.

In this paper a performance and cost model, based on the design of a generic router, is presented. Section 2 explains the need for a more flexible model of router design for performance analysis purposes. Section 3 describes the architecture of a generic router. An overview of the model is provided in section 4. In section 5 the set of router modules is introduced. Section 6 discusses the effects of design issues on the modules and discusses, in more depth, a selection of the modules. To illustrate use of the model, a model of the Torus Routing Chip is developed in section 7. Ongoing and future work is presented in section 8 and finally a summary is given in section 9.

2 Background and motivation

Design of a hardware router involves decisions regarding many interdependent design issues, selection and design of the required components, and design of their interconnection.

Evaluation of interconnection network performance using existing or proposed router designs involves development of specific models to meet the evaluation criteria and the type of router to be modelled. Due to the complexity in router design, these models involve simplifying assumptions for non relevant router components. This results in a restricted use of a model to the evaluation criteria in the specification. It also makes direct comparison of results from one model to another rather ambiguous.

One such assumption, often seen in the literature, is to assume a fixed delay through a router independent of its complexity [1]. However, it is shown by Chien [2] that router complexity has a significant effect on the delay encountered by a message traversing a router and, as such, is an important factor for performance analysis purposes.

Chien [2] developed a parameterised cost model, based on a specific technology, for a generic router architecture. However, although this generic architecture allows a number of routing algorithms to be compared, the focus on routing algorithm analysis removes the requirement to model other components in the router which have no effect on this analysis.

The choice of routing algorithm is one of many design issues. Chien[2] states that no single router architecture is suitable for comparison of all routing algorithms. Even more so, no single architecture allows comparison of a number of design issues.

A flexible model is therefore, required which offers cost and performance analysis of a wide range of routers with respect to various designs issues. This is the goal of this research.

3 The router architecture

A router may be considered as consisting of three logical parts, as shown in figure 1. The central part implements the functionality of the router — switching, routing and flow control. The switching method describes the physical path setup i.e. end-to-end or node-to-node, the routing algorithm decides which node is the next node i.e. which outgoing link to choose and the flow control checks the availability of the required buffers. The router applies this functionality to traffic entering and leaving at the two outer parts — the processor and the network interface.

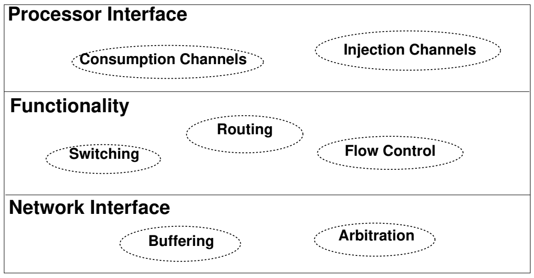

Figure 1. A functional router description

The structure of a router can be described as a collection of components working together to meet these functional requirements. For a specific router, the selection of the required components, the complexity of the individual components and the method of connected them is dependent on the design choices made. Complexity in router design arises from the variety of design issues and their interdependencies.

4 Overview of the model

The goal of the model is to provide a flexible router framework for modelling a wide range of router designs. Figure 2 illustrates the four key parts of the model - a set of generic modules, technology data, a set of parameters and performance analysis equations.

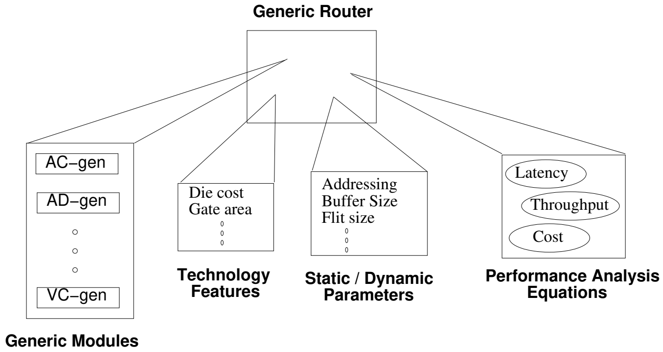

Figure 2. Network model

It is chosen to model the generic router as an interconnection of a set of modules. The set of modules chosen are a restricted subset of the generic modules available in the model. To limit the number of modules required, a module is not necessarily a single router component but may be made up of a number of components.

The modules are generic in that each module is not restricted to a particular implementation. Each module provides a frame into which a specific design, selected from the available module design set, is included. The framework includes a list of parameters effecting the design of the particular module and mathematical expressions which, when executed, provide performance data from the module to the generic router.

To provide flexibility in the model, many router features — referred to as decision issues, are provided as parameters to the model. Static parameters provide information regarding design choices made and dynamic parameters allow traffic patterns to be fed into the model.

Performance analysis equations consist of expressions for the three key performance features: latency, throughput and cost. Mathematical expressions to support these features depend on the number of parameters given to the model and as such, a selection of expressions are available to support different router issues. Technology data, expressing technology features, also provides the model with the ability to conduct cost analysis.

5 Identifying a set of router modules

A survey of a number of router designs was undertaken to build up the set of router modules— [4, 7, 6, 8, 5, 3]. These routers were chosen to represent both deterministic and adaptive, synchronous and asynchronous designs. It is assumed, that only point-to-point communication takes place in the network and therefore all multi-cast and broadcast features of these routers are ignored, at present. The resulting set of modules is shown in table 1 and the relationship between this set of modules and the functionality of the router, discussed in section 3, is shown in table 2.

Symbol	Module
AC	Address Comparator
AD	Address Decrementer
AR	Arbiter
AX	Address Converter
B	Data buffer
CB	Crossbar
CC	Consumption Channel
CS	Control Code Manager
CU	Channel Update
HU	Header Update
IJ	Injection Channel
MB	Message Build
MS	Message Split
MX	Multiplexor
RD	Routing Decision
SC	Signal Converter
SM	Signal Manager
SY	Synchroniser
VC	Virtual Channel Controller

Table 1. The set of router modules

The operation of many of the generic modules are self evident from their names and therefore a further description is not included in this paper e.g. address decrementer, address converter and data buffer.

The name address comparator has been chosen to represent both a zero checker and an address comparator due to the similarity of implementation.

Queueing in the router occurs both at the processor and the router interface, as described in section 3. However, it has been chosen to create separate modules for the network interface queues, the queues to the processor (consumption) and the queues from the processor (injection). This decision provides additional module structures at the interface where communication functionality, not currently implemented at the network level, may be implemented. As such, these modules may offer more than just the existing queueing function.

Function	Components
Processor Interface	IJ, CC
Switching	AR, CB, MX
Routing	AC, AD, AX, CU, HU, RD
Flow control	CS, MB, MS, VC, SM
Network Interface	B, SC, SY

Table 2. Functionality of the router modules

Switching in the router is often achieved by a crossbar **CB** and controlled by it's associated arbiter. A Crossbar module refers to the case of an $n \times n$ crossbar unit. However, if the switching functionality is distributed over the router to the extent that several $n \times 1$ units are required, then multiplexor modules are used instead of crossbar modules.

There are three levels of arbitration which may exist within a router. The first two levels, the **AR** module (crossbar arbiter), and **VC** module (virtual channel controller), allocate resources within a router. The third component, the **SM** module (signal manager), controls allocation of the external channel to the neighbouring node. There are two main reasons for separating the two internal arbitration functions. The first being their physical separation in most current routers. The second, is the difference in size of the arbitration units. The arbiter (**AR**) arbitrates between requests for output channels on a packet basis whereas the virtual channel controller (**VC**) may arbitrate either on a packet or flit basis.

Other more specialised modules include message split and message build which refer to the additional hardware required for the case where the control and data paths through a router are separated. Header update and channel update refer to hardware required to support adaptive routing and circuit switching respectively.

The virtual channel controller module includes both queueing and control logic to support all the virtual channels on a physical channel. As such, the buffering module is only required in routers where there are no virtual channels or there exists intermediate buffering within the router.

The two main methods of control implementation are request/acknowledgement lines and control codes. Being two quite different control mechanisms, it is chosen to implement these as two different modules — **SM** and **CS** respectively.

6 Design issue effects on router modules

As stated earlier, router design involves a large number of design issue decisions, some of which are interdependent.

Different design issues effect different components and traffic characteristics effect the components which exhibit decision or queueing features. Thus a generic router requires not only a set of modules to select from, but also a set of implementations (design set), for each module.

There are two types of design decisions — those affecting the router module itself (router parameters) and those affecting individual modules (module parameters). The router parameters e.g. switching technique, affect the performance expressions built into the model. and the number of instances of a given module required in the design. Module parameters on the other hand affect the design and number of instances required of a given module.

A full discussion of all the modules listed in table 1 is not possible within the space provided and therefore, taking the Torus Routing Chip [5] as an example, all further discussions refer to the models used in the description of the TRC. This set of modules includes the address comparator, address decrementer, arbiter, crossbar, routing decision unit, signal converter, signal manager and virtual channel controller.

6.1 Address comparator (AC)

The address comparator module represents either a zero checker — where relative addressing is implemented, or an address checker — for absolute addressing. It is often placed at the input to a router where it checks if the incoming message is destined for the local processor or not. The complexity of an AC depends on the length of the word to be compared. In this case, it is the length of the address or in other words the flit size.

In general, the number of AC modules required depends on the number of input channels to the router as a separate logic path is provided for each input. However, in the case of virtual channel deadlock avoidance, either the virtual channels on a particular physical channel follow the same logic path or a separate logic path is available for each virtual channel. As such, in the later case, the number of AC modules required is determined by the multiplication of the number of physical channels with the number of virtual channels per physical channel. Otherwise, the number of modules required is equal to the number of physical channels.

6.2 Address decrementer (AD)

An address decrementer is used in deterministic routers with relative addressing to decrement the relative address. Implementation of an AD is often achieved by some form of adder. The size of the adder is dependent on the flit size, where a flit holds the relative address in the current direction. As such, the design of the decrementer AD is parameterised by the flit size. The number of decrementers required is equivalent to the number of ACs i.e. the number of data paths in the router.

6.3 Arbiter (AR)

The purpose of an arbiter is to make a decision about the assignment of a shared resource. This decision is based on both the arbitration policy or method and the routing algorithm. The arbitration policy describes the priority of the competing inputs and the routing algorithm limits the choice of outputs for a given input. The delay through the arbiter is due to both the amount of traffic competing, the ease of winning (arbitration method) and the choice of outputs (routing algorithm). The complexity of the implementation is also dependent on whether control of the router is synchronous or asynchronous.

As such, the design of the arbitration unit is dependent on the arbitration policy; the number of channels to be arbitrated between; asynchronous or synchronous control; traffic in the network and the routing algorithm.

In theory, a single arbiter could be used to control the switching unit. However in practice, due to bottlenecks resulting in a central control unit, a form of distributed arbitration is usually implemented. One such form is where there exists one arbiter unit for each switching unit. Another form, as in the case described in section 6.4, the arbiter is further distributed and a small arbitration unit exists at each crossing point of the crossbar i.e. n^2 arbitration units. This is the case for the Torus Routing Chip.

6.4 Crossbar (CB)

As stated earlier, a crossbar unit handles part or all of the switching functionality of a router with the aid of an arbiter — see section 6.3. In general, for an $n \times n$ crossbar where there are n inputs and n outputs, 2n buses are required for the input and output data and n^2 crossing points for switching between these buses. Therefore, implementation is dependent on the number of inputs to the crossbar, equivalent to the number of data paths traversing it, and the size of the arbiter itself.

The number of crossbars required depends on the distribution of the crossbar in the router but will always be less than the number of data paths through the router.

6.5 Routing decision (RD)

The routing decision module includes a routing table which implements the routing algorithm. For deterministic routing, generation of the output channel is based on information regarding the input channel and the destination address. However in adaptive routing there are a choice of outputs for each input. The number of choices depends on the amount of adaptivity in the routing algorithm. The more adaptivity, the more complex the routing module. Traffic information supplied by the other routers is used to update each router's routing table such that a best output channel is selected based on the updated information. As such, complexity is determined by the routing algorithm and the number of outputs, or in other words the network degree. The number of routing modules is equivalent to the number of routing paths.

6.6 Signal converter (SC)

The signal converter handles convertion between two different signalling conventions. In the case of the TRC, 2-phase signalling is used between neighbouring routers and 4-phase signalling within the router. The complexity of the module is therefore dependent on the number of signal lines to be converted and the technology of implementation. The number of units required equals the number of output channels.

6.7 Signal manager (SM)

Control of the flow of data between two neighbouring routers may be achieved by request/acknowledgement lines or control codes. A sender must have control of the channel, the receiver must be ready to receive and in the case of virtual channels, a virtual channel must have won use of the channel before a request can be raised and data sent.

Generation of request/acknowledge signals is under the control of the signal manager (**SM**). There are two main signalling conventions in use. Either a single req/ack line may be used where the acknowledgement cancels the request or two separate lines are used where a request is acknowledged when both lines are equal. If virtual channels are implemented then the number of signal lines required is multiplied by the number of virtual channels per physical channel.

The complexity of the signal manager is thus dependent on the number of signal lines per physical channel which is in turn dependent on the number of virtual channels and the signalling convention. The number of units required is equal to the number of output channels.

6.8 Virtual channel controller (VC)

The Virtual channel controller includes both queueing for and multiplexing between the virtual channels of a physical channel. In the case of virtual channels it is more common to use output buffering than input buffering so that data destined for a particular virtual channel is not blocked behind data for other virtual channels on the same or different physical channels.

The parameters to this component are therefore the number of virtual channels per physical channel, the amount of output buffering provided and the arbitration method to arbitrate between the virtual channel queues. The VC module controls all the virtual channels for a single physical channel and therefore the number of VC modules required depends on the number of physical output channels of the router.

6.9 Summary of design issues

The effect of design issues on the design (**d**), number of (**n**), or performance (**v**) of a sub-set of the router modules is summarised in table 3.

	Modules							
Design Issues	AC	AD	AR	CB	RD	SC	SM	VC
Addressing	d							
Arbitration method			d					d
Buffer Size								d
Control			d			d	d	
Flit size(bits)	d	d						
Network Degree	n	n	d	dn	d	n	n	n
Number of VCs	n	n	d	dn	d	d	d	d
Queueing								d
Routing Algorithm			d		d			
Technology	d	d	d	d	d	d	d	
Network Traffic	f	f	v	f	fv	f	f	v

Table 3. Relationship between modules and their design issues

It should be noted that although traffic is not, in general, regarded as a model parameter but a model stimulator, it is included here as a parameter since traffic is fed into the implemented model as a dynamic parameter. Traffic does not directly effect either the design or the number of modules but the performance of a given design. The table shows the effect of network traffic as either fixed **f** or variable performance **v**. This implies that the performance of the **f** modules are unaffected whereas the performance of the modules with **v** vary with the amount of traffic in the network.

As would be expected, the choice of implementation technology effects the design of all modules. Symbol variants used in the table include **dn** and **fv**. **dn** refers to the discussion in section 6.4 and 6.8 regarding the extent of data path partitioning, which can give rise to either an increase in design complexity per unit or an increase in the number of units. In the case of deterministic routing, performance is independent of traffic **f**, and in adaptive routing performance is dependent on traffic **v** thus giving rise to the symbol **fv**.

7 Use of the model

Generating a router configuration requires four key components as shown in figure 3. A design file is created where the required modules are selected and their interconnection described. Default parameters may be chosen or actual parameters given to the design by updating the data files.

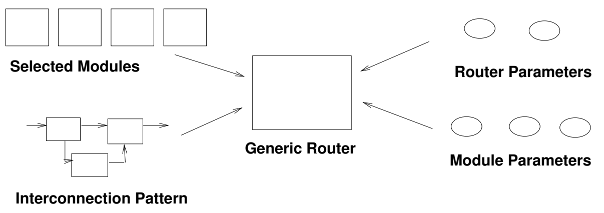

Figure 3. Choosing a router configuration

As an example, we will continue our discussion focusing on the Torus Routing Chip [5]. As described earlier, the list of required modules are: an address comparator, address decrementer, crossbar, arbiter, virtual channel controller, signal converter, signal managers and a routing decision module. This selection is specified in the design file along with the required interconnections.

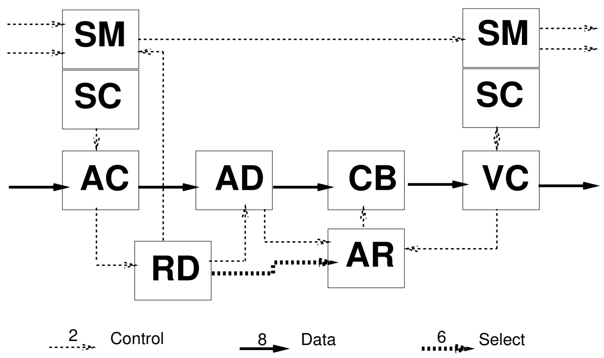

Figure 4. Torus Routing Chip - data and signal path

Figure 4 illustrates the interconnections required in the design file for a single path in the TRC. For each instance of a module, all incoming and outgoing signal and data connections are specified in respect to the adjoining module(s). An outgoing connection can be to a single or multiple destinations.

The next stage is to update the parameters for the design choices. Table 4 lists the relevant parameters for the TRC. These parameters have been chosen to represent TRC routers in a 4×4 mesh topology using a 10 flit packet.

Addressing :	relative
Arbitration - arbiter :	priority chain
Arbitration - VC :	fair
Buffer Size :	4 flits (32 bits)
Flit Size :	8 bits
Network Degree :	4
Network Diameter :	6
Number of VCs :	2
Packet size :	10 flits
Queueing :	output queueing
Routing Algorithm :	dimension order routing
Switching Technique :	virtual-cut-through
Control :	asynchronous, req/ack
Implementation technology :	3 μ CMOS

Table 4. Torus Routing Chip (TRC) design parameters

As stated earlier, these parameters are used to select the actual designs from the set of designs available. If we take for example the address decrementer, then assuming the design set shown in figure 5, the parameter list will result in the selection of the design AD_dsg_A_8. This is an 8 bit asynchronous arbiter. As there is only one technology available then the technology choice has no effect. At present, only one instance of a module will be automatically chosen by the parameter list, thus leaving the designer to specify the number of instances required. However the relevant parameters, in this case Network Degree and Number of Virtual Channels, are present to allow this feature to be added at a later date.

The router parameters, on the other hand, select the performance equation, in this case expressions for virtual-cut-through routing, and specify the packet size, header size and network diameter. Lastly traffic is added to simulate a loaded network.

8 Ongoing and future work

The model is currently in the implementation phase. The generic router and associated generic modules are implemented in the Performance Description Language (PDL) [9]. PDL is designed specifically for generic performance modelling enabling design instances to be compiled and analysed, thus suitable for this type of work.

The individual modules are being implemented in synthesisable VHDL code. Synthesisable VHDL code is chosen since the code may be synthesised into more than one

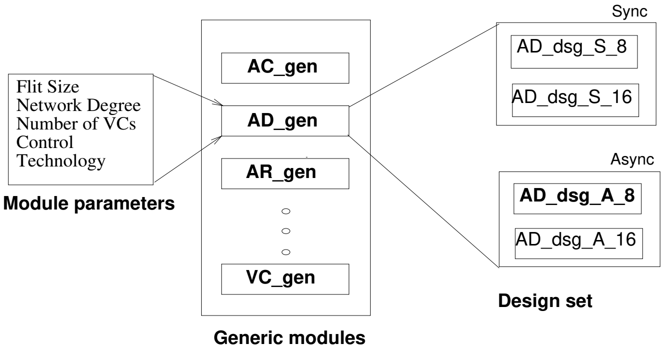

Figure 5. Relationship between parameter choice and module design

technology and may easily be adapted to reflect simple design changes. Therefore, it provides flexibility in design, a goal of this work.

Each net-list generated from synthesis of VHDL code is then converted into a PDL-compatible net-list. These net-lists will provide a library of component designs available to the PDL model.

9 Summary

This paper has described work which establishes a connection between conflicting design issues and the complexity of the routers final implementation. It therefore provides a base for modelling a wider variety of routers than current network level models provide and provides sufficient detail for performance cost trade-offs to be applied.

It is not assumed that the set of modules provided is the exhaustive set for all current and future routers. However, as the model assumes a router is built from a subset of the set of modules, a new router feature may be modelled by adding a new module to the module set. Also, as each individual component is parameterised by its respective design issues, these parameters can be varied to reflect changing design ideas. However, if a suitable design is not available in the design set then the respective VHDL code will have to be adjusted, synthesised and converted before the design will be available.

The range of available technologies are currently limited but may easily be expanded by including further data files for new technologies or updating existing data files to reflect advances in existing technologies. However, different technologies will often give rise to different net-lists and, similar to the above, new designs will require to be added to the library through synthesis and conversion of the VHDL code.

10 Acknowledgements

I would like to thank Lasse Natvig for his continuing support and advise in this ongoing project.

References

[1] A. Agarwal. Limits on Interconnection Network Performance. *IEEE Trans. on Parallel and Distributed Systems*, 2(4), Oct. 1991.

[2] A. A. Chien. A Cost and Speed Model for K-ary N-cube Wormhole Routers. In *Proc. of Hot Interconnects '93*, Aug. 1993. Dept. CS, University of Illinois at Urbana-Champaign.

[3] A. A. Chien and J. H. Kim. Planar-Adaptive Routing: Low Cost Adaptive Networks for Multiprocessors. In *19th Intern. Sympos. on Computer Architecture*, 1992. Dept. CS, University of Illinois at Urbana-Champaign.

[4] W. J. Dally and L. R. Dennison. The Reliable Router: A Reliable and High Performance Communication Substrate for Parallel Computers. In *Proceedings of the 1st International Parallel Computer Routing and Communication Workshop*, 1994. AI lab, MIT.

[5] W. J. Dally and C. L. Seitz. The Torus Routing Chip. *Distributed Computing*, 1:187–196, Jan 1986.

[6] W. J. Dally and P. Song. Design of a Self-Timed VLSI Multicomputer Communications Controller. *Proceedings of the Intern. Conf. on Computer Design*, (IEEE CS Press, 2473):230–234, 1987.

[7] J. M. Hsu and P. Banerjee. Hardware Support for Message Routing in a Distributed Memory Multicomputer. In *1990 International Conference on Parallel Processing*. University of Illinois at Urbana-Champaign, 1990.

[8] P. R. Nuth and W. J. Dally. The J-machine Network. In *IEEE International Conference on Computer Design: VLSI in Computers and Processors*, Oct. 1992.

[9] B. Vemuri, R. Mandayam, and V. Meduri. Performance Modelling using PDL. *IEEE Computer*, pages 44–53, April 1996.