Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

NETWORKING ON VLSI

Frank Schaffa
Mario Gerla

April 1990
CSD-900008

Networking on VLSI

Frank Schaffa and Mario Gerla
UCLA Computer Science Department

April 1990

Abstract

Advances in technology are making it possible to apply networking concepts to VLSI interconnection. This provides a framework for alternative solutions that are necessary due to the increasing delay of the interconnection in relation to gate delays. This approach is necessary so that high performance of increasingly more complex systems is achieved.

In this work we analyze the constraints imposed by the VLSI environment and propose an approach, which is based on strategies that simultaneously minimize the interconnection delay and increase the total throughput. The solution is specifically tailored to the given design constraints, which are quite different from conventional communication network constraints. The access protocols are discussed in terms of different characteristics such as liveness, flow control, fairness, and buffer requirements. Results based on simulation are presented. These protocols have no central control, which makes them suitable for VLSI circuits, e.g., the access mechanism is distributed with inherent flow control.

The results obtained in this work can be extended to high speed packet switching networks that integrate voice, video and data communication. The thrust is the use of streamlined protocols with low overhead to achieve low packet delay and overall high throughput.

1 Introduction

Advances in VLSI technology will soon change an important design paradigm, where the delay bottleneck will move from the active devices (gates) to the interconnection [1, 2]. At the same time, interconnection is becoming more dominant in terms of area usage [3]. Therefore, for high performance integrated systems it is important to find new interconnection alternatives that addresses both propagation delay costs and VLSI area costs.

Another motivation for this work is the difficulty of implementing large clocked systems [4]. The reason is the inevitable problem of clock skews and delays, which can be especially acute in VLSI systems as they become larger and denser.

In an attempt to solve this problem, Anceau [5] presents a synchronous scheme based on a distributed system organization, where each module has its own fast clock (each module is within a small isochronic region). It also communicates synchronously (via a bus) with the other modules, but at a slow clock speed. Metastability problems are avoided by preserving the clocks in phase. The obvious drawback of this approach is the slow communication capability because of the low speed of the bus and the contention for this single slow broadcast bus. This result reinforces our proposed system framework of independent modules that are interconnected to others via a network with the objective of optimizing throughput and minimizing delay. The network operates independently of the modules. This distinct functionality assists the hierarchical design approach, which is specially desirable in VLSI.

Addressing the interconnection problem, Mazumder in [6] presents an interesting evaluation of three types of static interconnection for VLSI with respect to three orthogonal criteria: physical (chip area and dissipation), computational speed (message delay and message density), and cost (chip yield, operational reliability, and layout cost). The three topologies considered were: binary tree, cube connected cycles and two dimension meshes. The choice of these topologies was motivated by their wide interest in the literature, as well as efficient VLSI layout (in terms of the AT^2 metric [7]), and representation of different topological classes. In this model, the average message delay is based on the average message path length. This measure, however, is accurate only if there is no contention for the links. Hence it is an optimistic lower bound for the delay. The average message density [6], defined as $\frac{N \cdot D}{L}$ (where N is the number of processors, D is the average path length, and L is the total number links), can provide some guidance in terms of bottlenecks. However, it is important to note that bottlenecks are very dependent on the traffic pattern, which is not considered there. Overall, we share the conclusion that topologies based on mesh structures are promising for VLSI interconnection.

Seitz [3] discusses the topologies for concurrent VLSI architectures in very general terms. His solutions are based mostly on algorithm communication requirements, without taking into account VLSI constraints.

Tewksbury [8] describes a sparsely connected mesh network for a large number of cells (modules) ($N > 400$) for a massive parallel multicomputer. However, while this approach might be interesting for WSI and wafer to wafer systems, it does not appear to be practical for VLSI due to the large amount of resources that might be needed just for the interconnection.

In this paper we present a novel VLSI interconnection approach based on a on-chip network concept. In Section 2, we discuss the impact of the technological advances based on scaling (namely, feature sizes getting smaller and chip size larger). We develop a delay model to understand the behavior of signals transmitted on long distance wires. This model is based on results taken from the existing literature and provides us with a better understanding of the delay performance due to stray capacitances and resistances. It also gives us the opportunity to observe how future technology, and its scaling down of component sizes, will affect the interconnection. Based on this model, we present a driving scheme for long interconnections that is capable of providing enough current to charge/discharge the stray capacitance, and also segments the wire with buffers, so that the wire resistance does not limit the charging/discharging current. An interesting property results from the segmentation, namely, the "pipelining" effect that occurs since buffers isolate one segment from the other. Hence, different information may reside in the line as it is being piped through, effectively increasing the throughput. These two factors, lower delay and the pipelining effect, motivate us to search for a networking solution for the long distance communication problem on chip.

In Section 3, we look into the specific requirements and characteristics of an on-chip network that is appropriate for the general VLSI design framework. We find that the most cost effective approach is a hierarchical design approach. The most significant advantage of this approach is the separation of the communication functions from the computation functions. This may facilitate resource sharing, logical reconfiguration, and pin count reduction. Implementation considerations lead us to assume a network characterized by a small number of switches, streamlined protocols and routing mechanisms, and small packet size (word wide, bit long).

In Section 4 we describe the chosen topology for this network and in Section 5 we discuss access protocols, first for a single ring and than for the proposed topology. These protocols are analyzed in terms of different characteristics such as liveness, flow control, fairness, and buffer needs.

Another important issue which impacts network performance, is the routing of information from source to destination. In Section 6, we present different routing schemes: adaptive (based on deflection) and; static (source routing as well as destination routing).

In Section 7 we present results based on simulation for the proposed network on chip. Finally, we summarise our contributions in Section 8.

2 Interconnection and Delay

2.1 Interconnection Scaling

The advances in VLSI process technology have lead to smaller dimension devices and at the same time to larger available areas for circuit integration. In order to understand how these advances will impact circuit design, a scaling theory was developed. A first-order MOS scaling model was introduced by Dennard et al. [9] and it is used throughout the literature [10] [11]. Basically it applies a dimensionless factor α ($\alpha > 1$) to all dimensions, voltages, and concentration densities. The larger α, the smaller the dimension.

Based on this first-order scaling theory for MOS technology, smaller dimensions imply faster transistors, i.e., switching delay scales down by α. Likewise, the number of transistors on a same chip size scales up by α^2. However, this only holds for small circuits. Performance of larger circuits is affected negatively by two factors: increased interconnection delay and increased interconnection area (wires), the former using up area that could be used for logic.

Parameter	Local Interconnection scale factor α	Long Distance Interconnection scale factor γ
Interconnection distance	$1/\alpha$	γ
Line resistance	α	$\alpha^2\gamma$
Line capacitance	$1/\alpha$	γ
Line response	1	$\alpha^2\gamma^2$

Table 1: Interconnection Scaling

Saraswat et al. [1] developed the MOS Scaling theory further to incorporate the effect of scaling on interconnection, classifying it in local interconnection and long distance interconnection. They take into consideration the scaling up of the usable chip area by a factor γ ($\gamma > 1$). For instance, line resistance on a "local" connection is $R = \frac{l}{w \cdot h}$ (where l is the length, w is the width, and h is the height of the wire), which, by the geometry scaling, becomes $R_\alpha = \frac{l/\alpha}{w/\alpha \cdot h/\alpha} = R \cdot \alpha$. However, on a "long distance" connection, line resistance becomes $R_{\alpha\gamma} = \frac{l \cdot \gamma}{w/\alpha \cdot h/\alpha} = R \cdot \alpha^2 \cdot \gamma$. Similar results can be derived for line capacitance. Line response (i.e.,time constant) is readily obtained as the RC product (Table 1).

Note that, as local interconnection scales down with α, its RC product remains constant, meaning that the time delay is the same, e.g. it does not scale down. The time delay problem is even more critical in long distance interconnection: signals have to travel longer distance because of the scaling down of the devices and also because of the scaling up of the chip size. Furthermore, the smaller devices handle less power, which limits their current driving capability. Hence charging/discharging the interconnection takes longer. These two scaling effects can be seen in Table 1. Figure 1 illustrates the interconnection scaling.

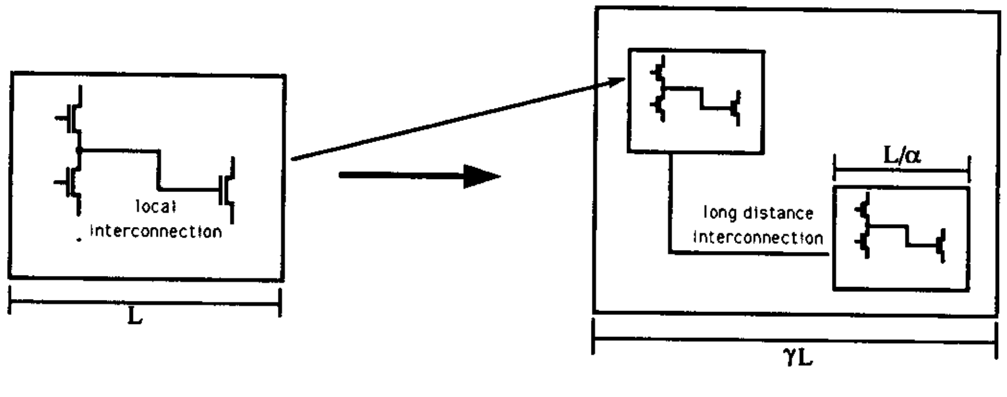

Figure 1: Interconnection scaling

In general, the effect of scaling in interconnection has been in focus in the literature over the past years. One of the most important results has been to show that interconnection delays will be of major concern in the submicron technology arena [12, 13, 14, 15, 1, 10]. To this effect, first order scaling provides a simplified model for the evaluation of the impact of the advances in VLSI technology. The model is reasonably accurate, and provides insight into issues that arise with the changes in technology. Note that the physical limit for the MOS technology is around minimum feature size of 0.1 μm with a useful area of 600 mm^2. This limit is expected to be reached by the year 2000 [16].

2.2 Interconnection Delay

The delay expression for the interconnection wire is based on an approximate result by Sakurai [17]. This result is based on an exponential approximation. For the range of parameters we will be using, the relative error is claimed to be in the order of a few percents. The approximate delay on an interconnection of length l is given by

$$t(l) = C_I R_I l^2 + 2R_{tr}C_{tr} + 2(R_I C_{tr} + R_{tr} C_I)l$$

where C_I and R_I are the capacitance and resistance per unit length of the interconnection, C_{tr} and R_{tr} characterize the transistor, and l is the interconnection length. Figure 2 illustrates the interconnection model, where R_i, C_i are the total interconnection resistance and capacitance.

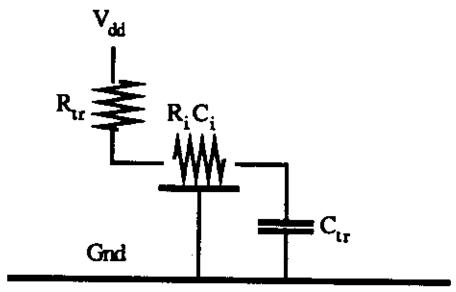

Figure 2: RC model

From the above equation, we note that in order to avoid the quadratic growth of t with l, we must break up the line into segments and buffer the signal (with so called repeaters) from segment to segment [18, 11, 19, 15]. This way, the delay grows approximately linearly with the number of segments, that is, with total connection length. Also, a high capacitance load (in this case the line itself) should be driven by a series of drivers of incrementally larger size which are capable of handling greater current [20, 21, 18, 15, 22, 23].

With this in mind, different driving schemes were tried using aluminum lines (metal wires) for the interconnection [24], in an effort to obtain the lowest possible delay for long distance on chip. Other materials like polysilicon and metal silicides have considerably worse performance [1, 25] due to the high RC constant and are therefore used only for local interconnection. The segmentation was the basis for the *S driver*[24]. The cascading of increasingly larger drivers motivated the *C driver* (capacitance driver).

The driving scheme with the best delay characteristic is a combination of segmented and capacitance driver (*SC driver*) shown in Figure 3. Similar results are reported by Bakoglu [15]. The line is segmented in equal size segments and each of them is driven by a capacitance driver buffer. These driver-segments are connected in series. The number of segments as well as the number of cascaded drivers is chosen so as to minimize the delay. The resulting delay expression is:

$$t_{SC} = k_s k_d f R_{tr} C_{tr} + \frac{R_i C_i}{k_s} + 2k_s\left(\frac{R_{tr} C_{tr}}{k_d} + \frac{R_i C_{tr}}{k_s} + \frac{R_{tr} C_i}{k_d k_s}\right) \tag{1}$$

where k_s is the number of segments and k_d is the number of cascaded drivers. R_{tr} and C_{tr} are the characteristics of the minimum size transistors, and R_i and C_i the total resistance and capacitance of the interconnection.

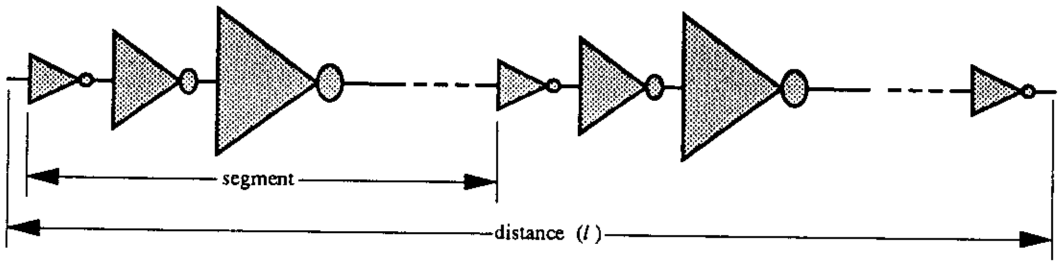

Figure 3: Segmented Capacitance (*SC*) Driver

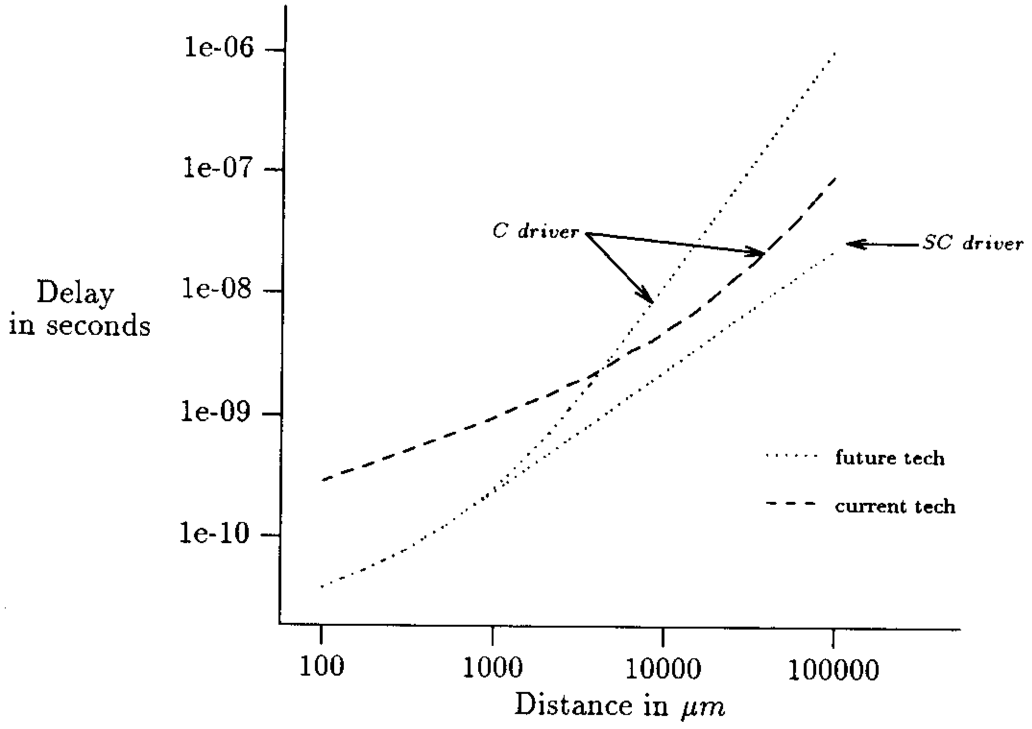

Figure 4: Delay versus distance

In Figure 4 we present results comparing the *SC driver* with the *C driver* for a .15μm technology (future technology). We also report for comparison the *C driver* results for the current technology since most bus interconnections are currently driven with *C drivers*.

The delay improvement yielded by the *SC driver* is due to the fact that this driver compensates for both the capacitance and the resistance of the line. Note that, as expected, the delay grows linearly with distance. Note also that C and *SC driver* delay are identical for short distances. The separation of the C curve and the SC curve occurs at the distance where the line starts being segmented.

2.3 Delay and Throughput

In previous works, the emphasis has been on the time delay that signals incur as they travel long distances on chip [15, 26, 1, 23, 22]. However, as important as the delay, is the throughput, since both factors affect the transfer of information. Therefore, it is not just a matter of how much time it takes to send information from one point to another, but also how much information can be sent in a given time interval.

One of our conclusions in the last section was that dividing up the long distance interconnection wire and buffering the signal at intermediate points, showed time behavior improvements. As a side effect, the partitioning also created "zones" of information, one zone per segment. Because of the buffering, the segments become logically isolated, so that different zones will carry different information at any given time. As exemplified in Figure 5, the packets of information move from one zone to the next, from source to destination like in a pipeline. This is the property that we want to exploit in order to improve the throughput. We should also note, however, that the segmented interconnection becomes unidirectional once buffers have been inserted (as opposed to the bidirectional nature of a continuous, bufferless wire which can broadcast signals in both directions).

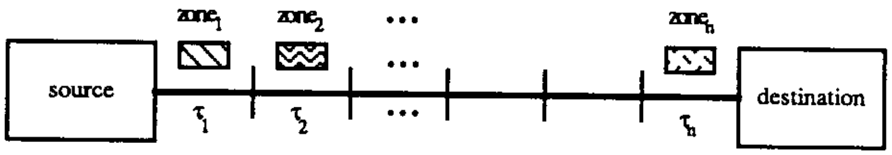

Figure 5: Zoning of a Interconnection Wire

In terms of throughput, the maximum throughput is the inverse of the highest segment delay. Trivially, one finds that for a given distance and fixed number of segments, the throughput is maximum when all segments have equal delay, i.e., equally spaced buffers assuming same electrical characteristic for all segments.

In our design, we wish to maximize the throughput and minimize the delay at the same time. While this is not always rigorously possible (in fact, as we shall see, throughput and delay taken separately are optimized by different values of numbers of segments), it is possible, however to come up with a combined performance measure which include both throughput and delay. The solution which we propose is the benefit function G:

$$G(s,l) = \frac{Tp(s,l)/Tp(1,l)}{Dl(s,l)/Dl(1,l)}$$

where $Tp(1,l)$ is the throughput over the interconnection with length l and with a single driver at the source. $Tp(s,l)$ is the throughput for the interconnection with s segments. $Dl(1,l)$ and $Dl(s,l)$ are the analogous for end to end delay. Of course $Tp(s,l)$ takes into consideration the pipelining effect.

Note that $G(s,l)$ is proportional to throughput and inversely proportional to delay. Hence, the benefit is high when the end to end delay is small and the throughput is high. Optimizing G will lead to a good operating point of the system, i.e., a good compromise between delay and throughput. A concept analogous to the benefit function G, known as "Power" [27] was developed for computer networks. Power is defined as the ratio of Throughput over Delay. Again, power optimization leads to an effective operating point of a communication link.

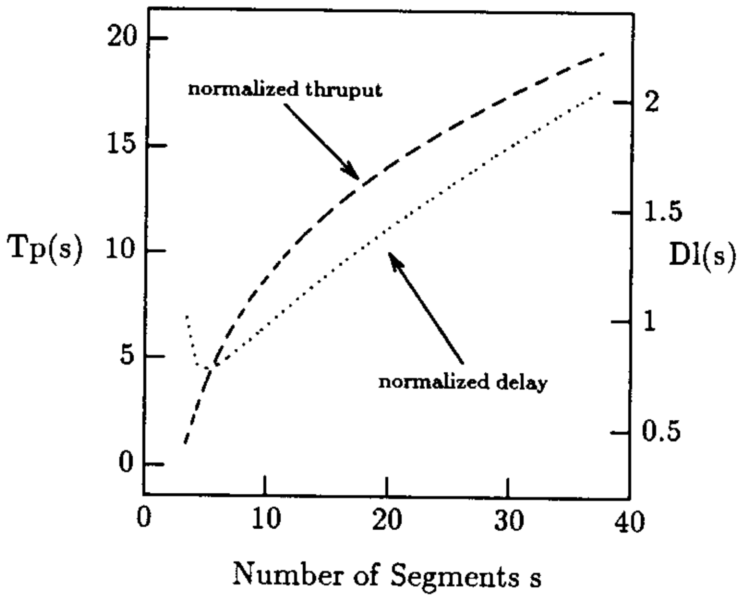

Figure 6: Throughput and Delay versus Number of Segments

The graph in Figure 6, shows the normalized throughput $(Tp(s,l)/Tp(1,l))$ and the normalized delay $(Dl(s,l)/Dl(1,l))$ for varying number of segments s and for an interconnection with length l, where l is the typical VLSI diameter. The delay values were derived from the SC driver expression in Eq.(1). The throughput per segment is the inverse of the delay of a segment. Note that, as the number of segments increases, the delay decreases until the number of segments approaches the optimal number of buffers (as seen in the previous section). From that point on, the delay increases because of the added delay of the buffers. The graph in Figure 7 shows the function G versus s. We note that for a small number of segments, the benefit function G is low because the delay is high (less segments than the optimal number) and the throughput is low (small number of zones). As we increase the number of segments, we cross the point were the delay is minimum. If we continue to increase the number of

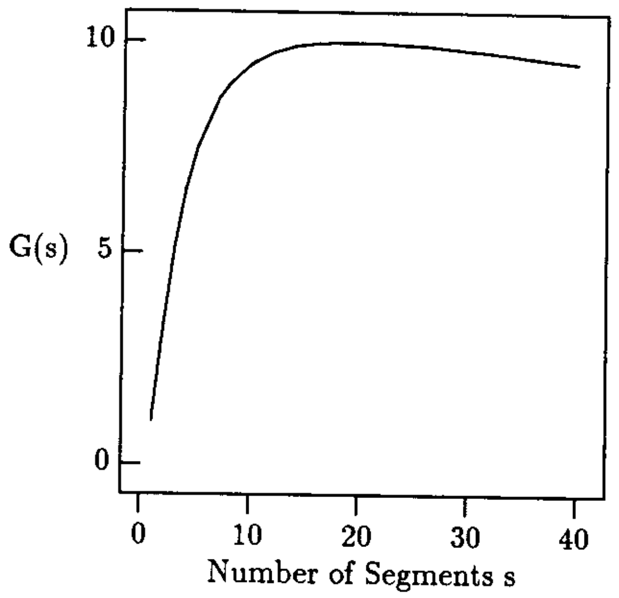

Figure 7: G versus Number of Segments

segments, we will increase the throughput and the delay. This means that after some point, the benefit of the increased throughput will be offset by the increased delay. The reason for the increased delay is that for each added segment, the buffer delay will add to the total delay.

We observe that the sensitivity of G is high for a small number of segments. This implies that there is a high gain even with a small number of segments. In terms of VLSI, this is interesting since more buffers mean more power dissipation and more occupied area and that a significant gain can be made with a small number of segments.

We should remark that G_{MAX} is not the absolute optimum for all applications, and that we should use the optimal G solution only as a convenient trade off between throughput and delay.

3 Networking on Chip

3.1 Motivation

Our choice of a network framework for chip interconnection was motivated by the throughput and delay results obtained on Section 2. In order to cope with the increasing line capacitance for long distance, and hence increasing delay, we opt for a solution which segments them. Signals traveling long distances have active elements (buffers) properly placed on the interconnecting wire. Two benefits of this solution are: 1) delay does not grow as dramatically, and 2) the *zoning* effect allows the coexistence of different information along the interconnection (pipelining). We will take advantage of both effects in our networking scheme.

VLSI chips are designed hierarchically due to the complexity involved; specifically, transistors are *bundled* into gates that in turn create functional blocks, which in turn are *bundled* into modules and so forth. At each increasing level on the hierarchy, more functionality is added. This is easily observed in practically all VLSI designs: microprocessors, memories, memory management units, floating point units, and many others.

The *bundling* is accomplished by the interconnection whose function is to transport signals from one transistor to another, from one gate to another, and so forth. The hierarchy that we see applied to the functional blocks, also applies to the interconnection (Figure 8), where each level has its own requirements.

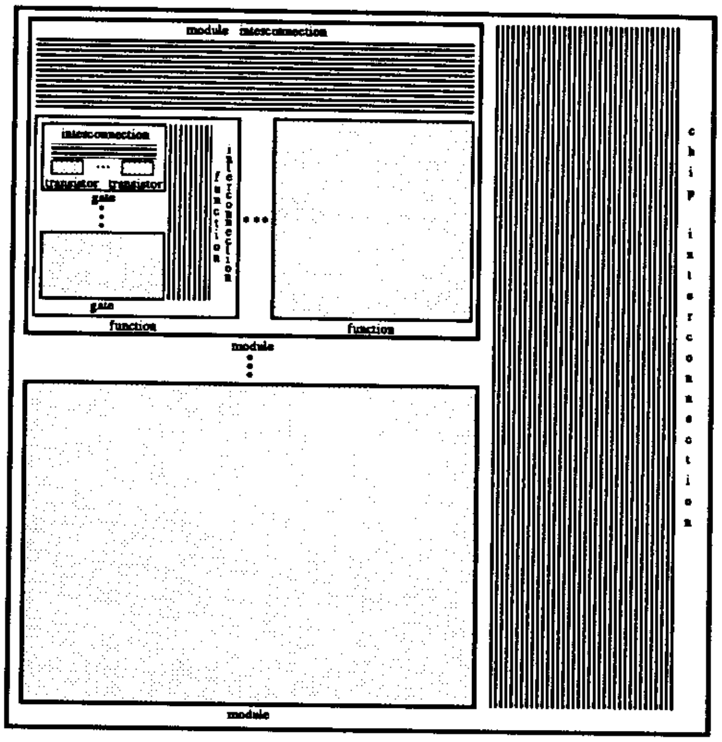

Figure 8: The Conceptual Hierarchy of a VLSI Chip

In the current technology, the highest level of intrachip interconnection hierarchy is the interconnection of modules. The next level is outside the boundary of the integrated circuit. The module interconnection has been traditionally accomplished by point to point interconnection and by bus based strategies. The bus communication strategies are very common and generally rely on a centralized model with tight control.

Given the technological advances discussed in Section 2, as the minimum feature size scales down (factor $1/\alpha, \alpha > 1$) and chip size scales up (factor $\gamma, \gamma > 1$) the gate density per area is increased, and so is the available chip area. In this context, modules from the current technology, which were at the highest level of the hierarchy will be *bundled* into module-clusters, and these clusters will become the next and last level of the on-chip hierarchy. Figure 9 shows this.

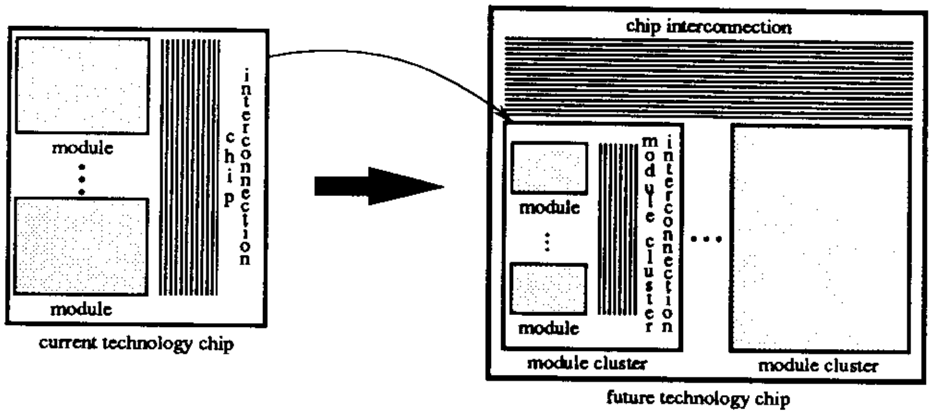

Figure 9: The Scaling of the Hierarchy of a VLSI Chip

To provide communication between these module-clusters we are proposing a networking scheme. We argue that networking on chip is a necessary step to keep up with the advances in the technology.

Another strong point of networking is the development of an interface between the intercommunicating blocks. This interface creates the separation of the communication functions such as routing and flow control from the blocks themselves. There are many advantages for such a separation:

- better control over the complexity of design, since hierarchy is enforced,
- standard communication interface,
- communication decisions done by the network elements, relieving blocks from such functions,
- interconnection area savings (due to channels sharing),
- flexible addressing,
- ability to integrate blocks that operate at different speeds.

One of the reason why networking has not yet happened in the current technology can be attributed to the fact that the interconnection delay becomes a significant problem only for very advanced technologies, which are still in the experimental phase. The gain for todays technology, if any, would be small because of the tradeoff between the area for modules and the area for the intra-chip network. This will change, however, as gate density as well as chip area increases.

3.2 Networking

An important, and often critical, parameter in the selection of the network architecture is network size. For VLSI applications, the size of the on-chip network will probably range from 16 to 64 nodes. Many tradeoffs must be considered to arrive at an appropriate number. Given that everything must be integrated on a certain area, a balance between number of modules and their sizes has to be achieved. If the number of modules becomes too high, the overhead of the network in terms of area (for circuits and wires) and power needs become high and modules have a lower functional integration (they are smaller). In the opposite case, it might happen that modules become big and sluggish because of internal delays and network resources inadequate to support efficient communications. This will negatively affect the overall chip performance.

It is important to make clear ar this point that a network architecture providing full interconnection (as the one we propose) is not the ideal solution for <u>all</u> intra-chip communication situations. For example, in architectures that can be classified as systolic arrays [28] other communication schemes will perform better. In that case communications usually occur only between neighbors, and there is no need for an effective and high performance global communication scheme, which is what we are aiming at. Our goal is a general communication fabric that can easily be used by VLSI designers.

3.2.1 On-Chip Networking Requirements

The networks that we propose here are very different from conventional networks such as LANs or WANs. The goal of our networks is to achieve global high performance on intra-chip VLSI communication. Therefore, a very specialized set of requirements (different from conventional networks) must be taken into account. We envision these networks as characterized by:

- assured delivery,
- streamlined protocols,
- fast routing decisions,
- small packet size (one bit deep, word wide),
- regular topology matching the two dimensions of integrated circuits.

Assured delivery implies that a packet, once accepted, will arrive to its destination within a bounded time (no loss, no buffer overflow). Protocols should ensure that there is no buffer overflow. Loss due to line errors is negligeble (perfect channel assumption). Loss due to failures is very unlikely, with probability smaller than the probability of failure of a module. (Recall that faults are a function of used area and switches are significantly smaller than modules [29, 30]). The motivation for such assumption is that it simplifies the design of the switches, making it feasible to implement them at this level of the hierarchy.

Protocols must be simple and efficient since, in order to achieve high performance, the logic decisions should be processed in the order of a few gate delays. Here the previous assumption of perfect channel comes very handy: no need for acknowledgement, time-out, and retransmission mechanisms. If those mechanisms were necessary, great amount of area would have to be dedicated. Besides, they would lower the response time of the protocol in two ways: circuits would be more complex and slower, and bandwidth and processing overhead would be introduced. Windowing mechanisms, common in other networks, are also prohibitively expensive it this domain. If error checking is deemed necessary, it should be implemented at a higher level, for example, process level.

The size of the packets should be small because at this level on the hierarchy the granularity of information is small. Large packets would mean higher use of the communication resource in terms of time and space (contention and buffer space). In our study we have chosen packets that carry one word of information (32bits) in parallel. This implies that links have 32 lines of data plus address and control lines.

The routing of packets from source to destination must be simple and efficient. The small packet size (one word) does not allow for any complex routing computation.

The topology issue is also very important: we should rely on topologies that fit nicely within the delay constrains presented in Section 2 and the planar characteristic of VLSI. Node distances should be dictated by the need of amplification/regeneration in order to attain the minimum delay.

Reliability considerations in this environment are also very different from what we find in conventional networks. In the LAN arena, for example, ring networks are usually criticized because of the unreliability of the serial active elements (switches) [31]. Special steps must be taken to provide fault tolerance. In our case these problems are not so critical since, as mentioned earlier, the area for the switches is small in relationship to the total chip area and faults are a function of area [29].

3.2.2 Performance Criteria

In order to select a proper design, we must first define a set of performance criteria.

The usual performance criteria are: delay, throughput, fairness, and cost. Delay is a measure of the time packets take to travel from origin to destination. It may relate to the average time or to the maximum time. Throughput measures the number of packets that go across the network per unit time. This was discussed in Section 2. Fairness refers to the equality of distribution of the bandwidth. Cost is related to the complexity and to the physical size of the design which determines its yield.

These criteria are not orthogonal. In fact, they may conflict with one another. For instance, high throughput (desirable) may lead to high delay (undesirable) and low fairness (undesirable); low cost (desirable) by using serial communication may lead to high delay (undesirable). Thus, a compromise must often be achieved. Also, additional constraints, which vary from application to application, may actually fix some of the measures (e.g. delay).

3.2.3 Design Parameters

As important as the performance measures are the network design parameters (i.e. design variables and constraints). Some of these parameters are unique of VLSI design and determine how the performance objectives may be attained. A partial list includes:

- *Routing mechanism*: How packets are routed from source to destination. The mechanism helps to alleviate congestion by deviating packets to less used paths.
- *Access protocol and flow-control*: How packets gain access to the network and how congestion is handled by slowing down the incoming traffic. By slowing down the traffic, delay decreases and fairness may improve.
- *Redundancy*: How redundancy and reconfiguration mechanisms may reduce manufacturing cost by increasing the yield or, may decrease the maintenance cost by prolonging the useful life of the system.
- *Layout*: Deals with how networks may be laid out on chip. There are a series of constraints since the basic space is a two dimensional one with a very limited number of independent layers that may overlap. The common value for these different layers is two or three. Some other important characteristics of this mapping are regularity, distance between switches, and the capacitance involved with this mapping. Care should be exercised to avoid complex layouts with wires crisscrossing; this would make it more prone to noise, crosstalk, greater capacitance, and also make it more difficulty for power distribution wires (power grid).
- *Number of ports per switch*: A large number of ports implies a more complex switch that needs more circuitry and consequently more area. This usually leads to slower switches with higher delay. Furthermore, more power would be needed for the extra load (logic and drivers), and an increased difficulty in the mapping, discussed above. Similar conclusion was reach by Fujimoto [32].
- *Number of ports per bus*: If a multiaccess, multiport bus is used, a larger number of ports involves higher capacitance on the bus since ports are capacitive loads. Hence, high driving currents are needed for faster transmission, implying larger drivers and larger dissipated power. The end result is an increase in delay through both, the driver chain and bus.

- *Height and width of wires*: Height and width of wires are parameters of the wire capacitance and resistance (see Section 2) and consequently parameters for the delay. However, the delay cannot go below some value because it is bounded by the maximum sectional current that may flow across metal lines. Currents above the limit produce metal migration [10, 12], which will literary carry metal away from the "hot spots" and eventually brake the connection. Using wider wires will increase the capacitance, which in turn will need more power (current). The height of the wire is limited by the technology. The "solution" lies in careful design.

4 Chosen Topology

Since grid structures (mesh) have very desirable properties regarding VLSI layout (regularity, spacing, 2 dimension layout, among others), the topology presented here is based on a square grid with jumper links added at the boundary to provide exactly $2 inputs\ and\ 2 outputs$ at each node (see Figure 10) [24]. We have named this topology MRing because of the multiple rings created by such interconnection. This topology satisfies some of the criteria presented earlier namely, regularity, number of ports, and layout.

At the intersection points of the grid are the switches, i.e. the nodes of the network. Blocks or modules (the building blocks at that level of the hierarchy) are located in the inner spaces of the grid and interface with the network through the switches (Figure 10).

Packets travel from switch to switch via links which are unidirectional. The directions of the rows and columns alternate between right to left and left to right, and between bottom to top and top to bottom respectively.

An interesting characteristic of this topology is that neighboring switches at the boundary are connected by two links in parallel. In order to distinguish the links one from the other we assign them to two separate planes. Referring to Figure 10, plane x (the orientation is decided by the elongation) consists by the rings formed by switches in the sets $\{1,2,3,4,8,7,6,5\}$ and $\{9,10,11,12,16,15,14,13\}$, while Plane y has $\{1,2,6,10,14,13,9,5\}$ and $\{3,4,8,12,16,15,11,7\}$.

The boundary links create more paths between source and destination pairs thus helping distribute the load evenly among the links. However, their presence also increases the complexity of the routing algorithm, as discussed in the Routing Section.

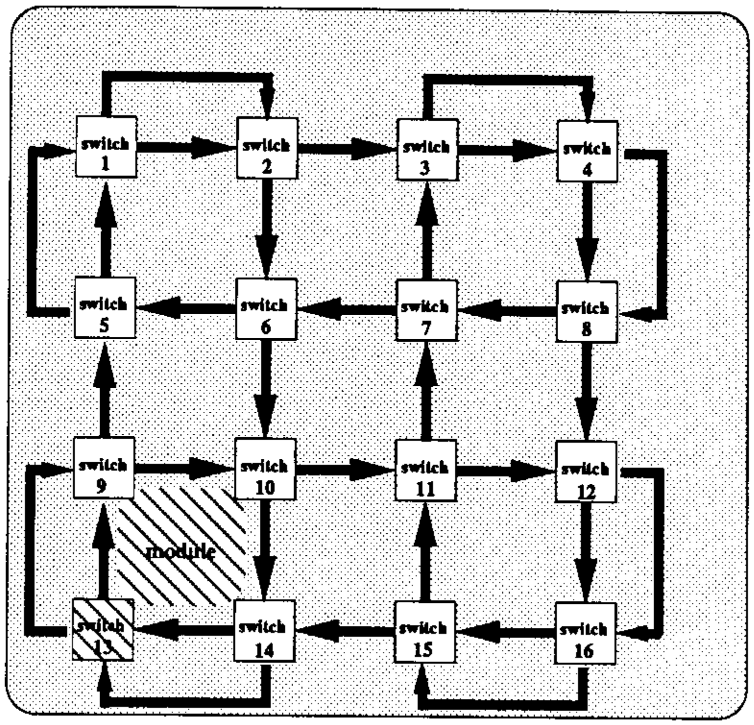

Figure 10: MRing Topology

5 Access Protocols

5.1 Access Protocols for Slotted Rings

In this section we discuss different access protocols for the MRing network. For simplicity, we examine first a single ring in the MRing arrangement (see Figure 11). Since our packets are of fixed size (i.e. one word) we may assume that the ring is a slotted ring. The slotted ring was first studied by Pierce [33] for bit serial communications. In our case however, we use a bit parallel link, that carries one word per slot. We cannot apply directly the slotted ring protocol, thus we must search for different solutions. The alternative approaches discussed here are of two types: token and token_less based protocols. The token schemes are intrinsically pessimistic and usually "protect" the system from the worst case situation. Token_less schemes, on the other hand, are in principle optimistic leading to lower delays at lower loads.

5.1.1 Token Based Access Protocols

We assume that N frames (i.e. slots) are maintained on the ring, where N is the number of nodes. That is, there is one frame per link. In the explicit token scheme, one of the control signals has the function of token indicator, and is marked only in one of the N frames, called the token frame.

A node with a packet to transmit, waits for the token to come by and then uses the first free frame thereafter. Frames are marked empty by the destination node. The frame carrying the token is used for data as any other frame. In heavy load, a node would have access on average to every N frames (round robin). On light load, the average waiting for the token is $N/2$ frames with many empty frames not being used. The result is high latency. This is why we consider the token scheme as a very conservative scheme. Namely, this scheme is fair at high as well as light loads; however, at light loads, it cannot take advantage of unused resources to increase throughput and decrease access time.

5.1.2 Token_Less Based Access Protocols

A major difference between token and token_less schemes is that in the former the access control information is conveyed by the network (via the token) while in the latter, the nodes use some internal state to control the access to the network. We have devised two token_less schemes: DIRC (distributed input rate control) and DIRC_BP (DIRC with back pressure mechanism)

In the DIRC scheme the input rate is independently controlled by each node. Namely, the access decision is based on the amount of time the packet has spent waiting. A counter keeps track of the waiting at each node. Initially or after each transmission, the counter is set to FC and is decremented with the passage of frames. When the counter reaches zero, the node may take the first free frame that passes by. Frames are marked empty at destination. If FC is set to $N + 1$ and the load is high this scheme behaves as the explicit token scheme. In light load, however, the wait for the turn to transmit is lower because nodes decrement the counter at each passing frame, independently of having a packet ready to send or not. Thus, a newly arriving packet may be able to go immediately. This scheme is detailed and analyzed in [24].

DIRC with Back_Pressure Protocol

In order to reduce the access time even further we developed DIRC_BP, where the basic idea is to let a node use the first free frame that comes by. This mechanism also increases the throughput significantly. Of course, precautions must be taken in order to provide a minimum service for all nodes, avoiding starvation, or deadlocks. In DIRC_BP, if a packet has waited more than C frames, it can request a free frame from the previous node (see Figure 11). The request for a free frame is satisfied by a node as the highest priority of service. Namely, in terms of service priority, from the highest to the lowest, we have a) free frame request, b) through traffic, and c) local traffic. If a node is holding a through (i.e. transit) packet when it receives the free frame request from its downstream neighbor, it must store the through packet in its buffer and release an empty frame. The node will, however, in turn request a free frame from its upstream neighbor, and so on. Thus, the free frame request propagates upstream, until it reaches a node with an empty frame. It is through this backpressure action (hence the name of the protocol) that local packets can be inserted into the ring after having waited C frames.

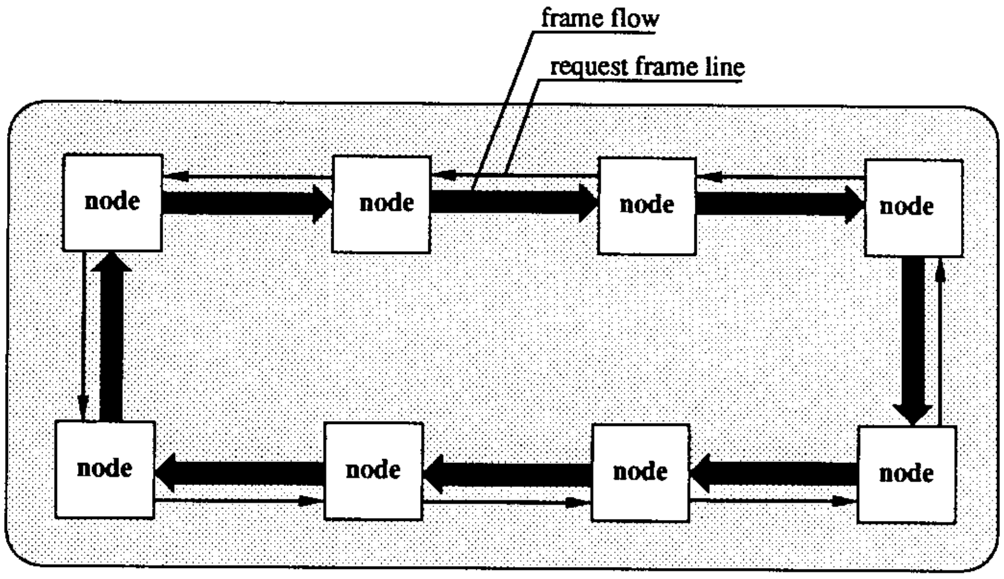

Figure 11: Cyclic Topology and Flow

At each node, for access control purposes, there is a *counter* and a logical flag *alt*. The *counter* has the function of keeping track of the waiting of a local packet for a free frame (before the counter expires, a local packet may enter the network only on free frames, no preemption). The flag *alt* is used to alternate the requests to the upstream node ($node_{i-1}$) when back pressure is to be applied. The protocol is described in Appendix A.

Back pressure, i.e., RF (request for frame) is applied when a passing frame is occupied and either (or both): *counter* reaches zero (because of a local packet waiting) or; a through packet was queued at this node ($node_i$) due to a request from the downstream node ($node_{i+1}$). Requests for free frame due to $counter = 0$ are sent on alternating frame times. The reason for alternating is evident from the diagram in Figure 12b. It takes two time ticks for a *node i* to get a free frame after the request. Thus, there is no need to repeat the request just after one tick.

Some Properties of the Basic Ring Access Protocol

This protocol is deadlock free with $C \geq N$. The proof consists of showing that no cycle for free frame requests is created [24]. In fact, a request propagates along the opposite

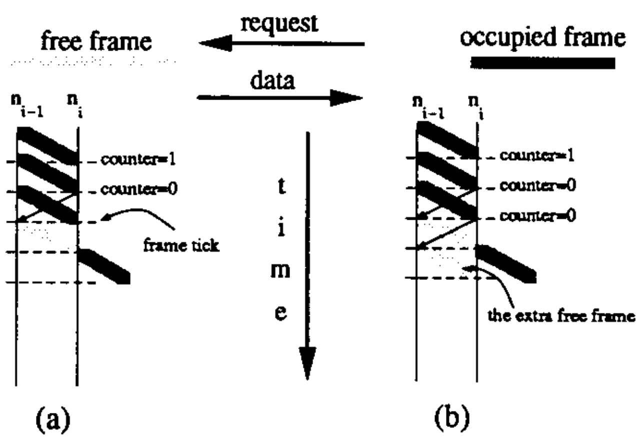

Figure 12: Timing and Flow of Data and Control

direction of the packet flow (upstream) till it finds a free frame (Figure 13). This will happen whenever the request reaches a packet that just arrived at its destination, or; it reaches a node with no packets (either local or through). Since with the $C \geq N$ condition there is always a free frame in the ring or a packet arriving at its destination, eventually the request will be consumed.

This protocol also allows for a non-uniform bandwidth allocation. This is done by tuning the C_i for the individuals nodes. The smaller C_i is, the more pressure that node exerts on the network, therefore getting more bandwidth. The condition for deadlock freedom becomes: $\sum_{i=1}^{N} C_i \geq N^2$.

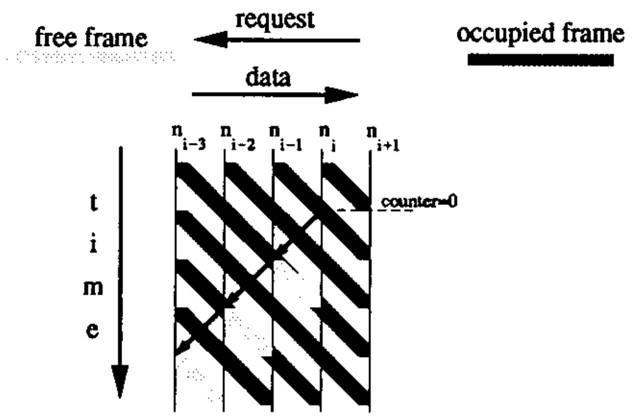

Figure 13: Flow of Free Frame Requests

Another interesting point about this protocol is that the buffer size of the through_buffer per node is bounded by *two*. This is because a node, as soon as it receives a request, forwards a free frame, queueing the through packet just coming to the node (in case there is a packet and the packet is not destined to this node). At the same time, this node requests a free frame from the previous node. The second position in the buffer is taken by the packet that left the upstream node as the request for a free frame got there. This is illustrated in Figure 14. By the time an eventual next request would be coming to this node, a free frame will be arriving (per its previous request). In conclusion, no additional packets will be queued at a node. The two packet buffer size limit is highly desirable in a small network, where resources are "precious."

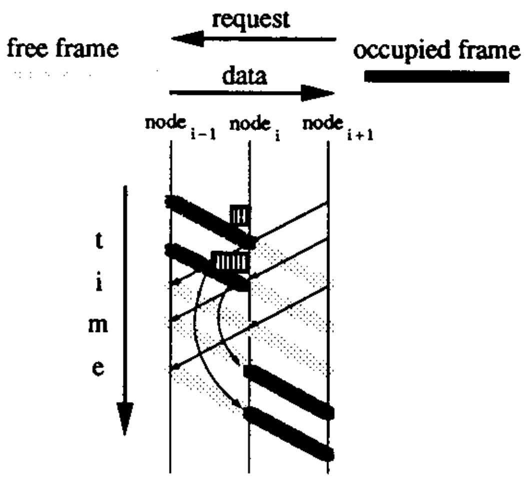

Figure 14: Two Buffers Only

Function of the Counter

In a heavy load, uniform traffic situation , and with the counter set to a high value ($C >> N$), a node will get a chance to transmit when a packet destined to it arrives, i.e., the packet is "consumed", freeing up the frame which is imediately taken by the local waiting packet. Under the $C >> N$ assumption, no node will have to defer "this" free frame since hardly any counter ever reaches zero.

The function of the counter is to distribute ("spread") the free frames generated by arrivals at nodes, so that nodes which are not popular destinations also get a chance to transmit. This is the basic function of the upstream frame request, that will force a frequently addressed node to defer transmission of its own and pass the free frame downstream to a waiting node.

Regarding the through packet queue, the request for a free frame can be changed so that it is triggered when queue is B rather than 1 as described before. This will allow a node to absorb a greater variance of traffic without requesting a free frame so often. This way, the number of free frames moving on the net is reduced, and the throughput performance could be improved. If requests are issued when the through buffer has B packets, it also follows that $B + 1$ buffers are required at each node.

Variations of the Basic Ring Access Protocol

In order to explore the tradeoffs between protocol efficiency and complexity of implementation, we have investigated several variations of the basic protocol. A simpler version: request for free frame when condition arises is only done in alternate frame ticks. This protocol is deadlock free for any C and the through_buffer size is bounded by *one*. However, starvation is possible in very unusual situations. A more complex version, which provides fair access even on unusual situations, is based on the idea of nodes getting credit for free frames they had to defer while they were waiting to send (counter $= 0$) and use this credit towards the next local packet. The details of these protocols can be found in [24].

5.2 Performance of the DIRC_BP Protocol

In this section we present performance results for the DIRC_BP for a 8 node unidirectional ring. We assume that the link delay is normalized to 1 unit of time, the destination address is based on the maximum distance from the source node (two cases: 7 or 4) and is chosen uniformly and independently, and exponencial interarrival time for new packets with waiting room equal to one.

In Figure 15 we present the results for the Total Packet Delay (from the arrival time to the time the packet reaches its destination) versus Throughput of the network, as the Interarrival Time for packets decreases. We consider two cases regarding the distance from source to destination. In the first one, packets can be sent to any other node; in this case, the maximum distance in a 8 node ring is 7 hops. In the other case, destination cannot be more than 4 hops away (packets stay less time in the network).

The Total Packet Delay is bounded since we have only one buffer for new packets per node and from Figure 15 we note the improved performance in both delay and throughput as the average hop distance decreases. This shows the efficient utilizations of the frames in the network.

These results were obtained from a simulation model and from a analytical model based on a markov chain. The details of the analytical model are in the Appendix B.

5.3 Access Protocol for the Mring Topology

We now direct our attention to the full, multiple ring topology. We recall that in the MRing topology (see Figure 10), a switch is positioned at each crosspoint and is reponsible to provide the overall coordination of the traffic. The switch consists of two nodes, each of which takes care of one of the planes as explained earlier (x for the loop elongated on the x direction and y for the other loop). A traffic flow diagram is shown in Figure 16.

Access coordination of all traffics (local, through, and switched) to the output link of each

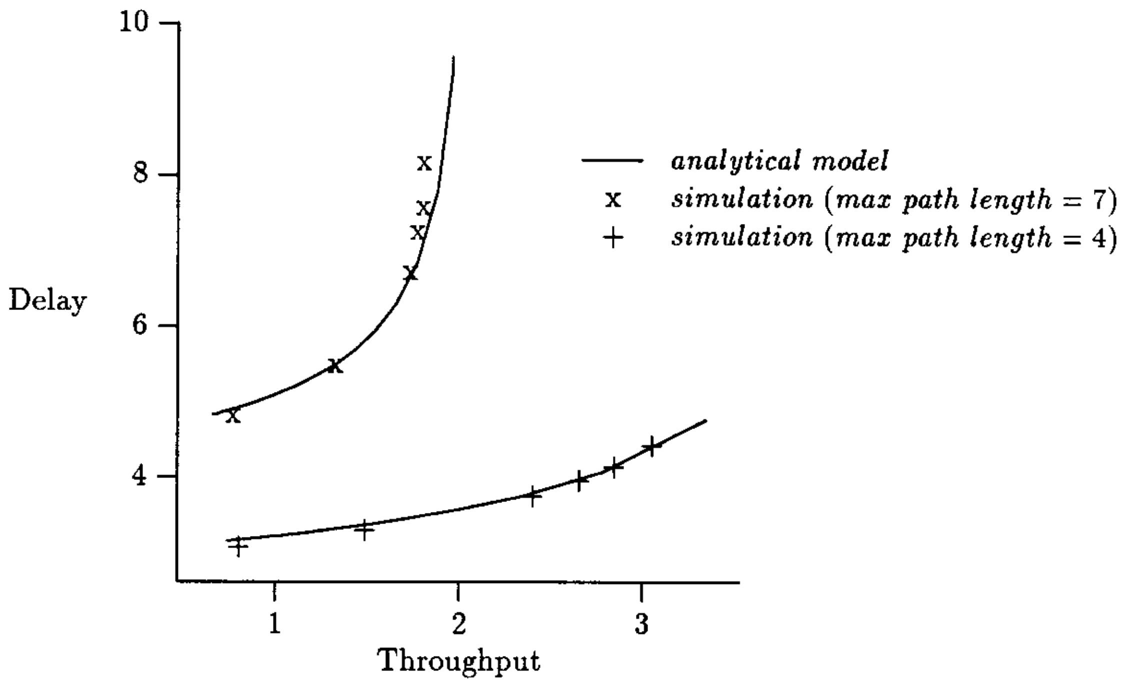

Figure 15: Performance Results for a 8 Node Ring

of the planes is performed by the node protocol. Namely, the protocol alternates the service priority to each of the through input links at each frame tick by alternating between mode x and mode y. This basically "reserves" half of the bandwidth a priori for traffic coming from x and the other half for the traffic coming through y. The local traffic gets access in the worst case when the counter expires. Figure 17 illustrates the concept of a node for this protocol. For instance, in mode x, the node first checks for through traffic in x. If any, it sends it. In case there is none, y is checked and, if there is through traffic there, it is sent. Finally, the node checks for local packets. After this, the protocol switches mode and checks first for y and so on.

Local traffic also alternates so that requests for free frames are issued to both incoming links. When counter reaches zero, the local packet is sent out on behaf of the link which matches the *alt_local* flag. This packet goes before any through traffic (since it has already waited the count period).

Back_pressure may be applied to both incoming links and is mode dependent. For example, in mode x if $queue(x) > 0$, a request for free frame is issued on $link_x$. Moreover, if $queue(y) > 1$, a request is issued on $link_y$. And similarly for mode y. The first test generates the alternate requests that guarantees the half of the bandwidth, while the second test makes the requests consecutive in case there is accumulation. At each node the maximum number of packets that can at any time be queued is four. The algorithm is described in Appendix C and simulation performance results can be seen in Figures 19, 20 in Section 7.

The above protocol may provide unfair service under certain conditions. For example, since the protocol is biased towards providing half of the frames to each of the incoming links, a $switch_A$ could use half of the frames for packets that entered the network on the

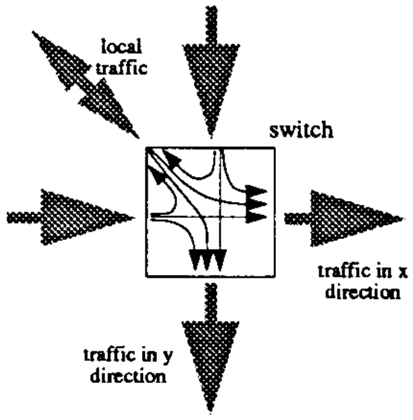

Figure 16: Flow of Data in a Switch

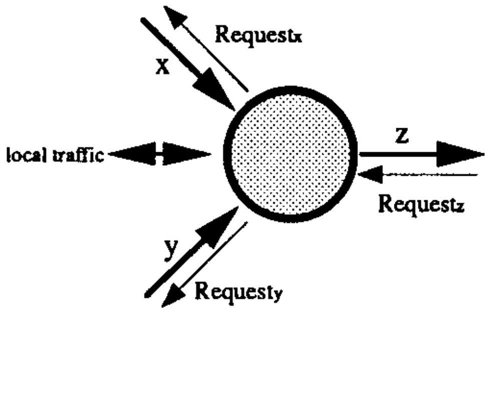

Figure 17: Node Traffic Flow

previous switch ($switch_B$) just one hop away (from one of the input links), and use the other half of the frames for the other input link. However, the packets on the second link could have been packets that waited longer in the network (i.e. that have been through merging waits in previous hops). Thus, a switch further away from $switch_A$ than $switch_B$ would be getting less bandwidth than $switch_B$ even though it may have the same local input rate as $switch_B$. One proposed solution is to add one more field in the link, which keeps track of the number of waits that the packet has been through since it was inserted in the network. The wait count is used to ascertain the access priority between the cotending input links (for through packets); higher wait count indicates higher priority. The basic idea of this scheme is to give higher priority to packets that have been longer in the network. This variant of the basic scheme was simulated. The results are similar to those of the original scheme for uniform traffic. They present a more fair allocation of frames for non uniform traffic.

6 Routing

At each switch, a packet may enter the subnetwork, leave, continue to the next switch in the same direction, or change direction (or plane). The actions related to these events are under the control of the routing function.

In our application, routing decisions must be made very fast. Our goal is to achieve packet delay from source to destination comparable to the signal propagation delay on a wire covering the same distance. There is no time for sophysticated routing processing while the packet is coming into the node. Hence, the routing computation is to be done off line or in the background and loaded into the nodes as load conditions change. We assume that changes in load patterns as well as changes in topology (due to failures) are infrequent when compared with packet transmission rates.

In order to perform the routing functions in a rapid way, two approaches have been considered: (1) Adaptive routing based on deflection and, (2) fixed routing.

The basic idea of deflection is to make the routing decisions at the local level based on immediate conflicts. The concept applies to networks with $2in - 2out$ switches and no nodal buffering; it was first presented by Maxemchuk [34, 35] and further analyzed in [36]. The routing decision is made based on the destination of the incoming packets. Since there is no buffering, output link conflicts are solved by "deflecting" one of the packets (i.e. the losing packet takes the output link which is not on the shortest path to the destination). This, however, implies that the local source is to transmit only when one of the outgoing links is not used by one of the incoming links [34]. Since fairness is an important issue, the local traffic should not depend on free frames that eventually might come by. In fact, switches have 3 inputs and 3 outputs, the third input being used by the local traffic and the third output by packets that reach their destination. In our case, since deflection guarantees that there is no backlog among the two major links, we may use any of the protocols for a single ring (presented earlier) for conflict resolution between the local traffic and the network traffic.

For fixed routing, two alternative mechanisms have been proposed. The first one (RT) is based on Routing Table lookup by destination (destination routing) at each switch. The second one (source routing) is based on shift-decide (SD)at each switch. In the first solution, at each switch, the packet destination address is used to access the routing table, which will be used by the switch to take the appropriate action. Therefore, the address on the packet needs $\log_2 N$ lines of the bus, N being the size of the network. Note that the same address length is required by the deflection mechanism.

In the second scheme, the path is encoded in the packet path field, (which is equivalent to the destination field in the other scheme) using $M + 1$ lines of the bus, where M is the longest path from any source to any destination. In the 4x4 network, $M = 8$. Initially the address field is loaded with the "path" to the destination on the top most bits and the rest is loaded with "$0 \cdots 01$." At each switch, the address is shifted out to a single bit register and a "0" is shifted in. The single bit information is used to make the routing decision. When the all-zero address detector is true, the packet has arrived to its destination.

The destination routing (RT) mechanism with some additional logic can also provide broadcasting of packets. A new signal on the links is needed to indicate whether the frame contains a broadcast packet or not. This signal is used to select the broadcast routing table. Some of these routing schemes are evaluated in Section 7.

So far, we have discussed the mechanism by which messages can be routed from origin to destination within the subnetwork. The problem of creating these routing tables as well as the details of the routing mechanisms are discussed in [24]. The algorithm to find the sub-optimal RT's is based on distributing the traffic (based on shortest path and load requirements) so that link utilization is as even as possible. Actually, there is a good reason to restrict the solution to shortest path routes: the number of hops from S to D in this topology is $4 \cdot k + d$, where d is the number of hops for the shortest path and k is an integer (including 0). Thus, the penalty for not using a shortest path is at least four additional hops. Note that in small networks like the ones proposed here, this penalty is relatively speaking very stiff; for example in a 4x4 grid the longest shortest path is 8 hops. This also explains why fixed routing performs better than deflection routing (as shown in the next section). In summary, adding 4 hops to a path would increase delay by a large margin, even when this is done to avoid congested links. Two reasons being: 1) the traffic would be diverted to a longer path and 2) it would contribute to congestion on each of the extra links.

7 Numerical Results

In order to evaluate our network approach, we modeled a 4 by 4 MRing topology, with the access protocol described in Section 5.2, and with deflection as well as fixed routing. Routing tables were generated using a heuristic algorithm that minimizes the differences between link utilizations [24].

We first present a comparison of the two routing schemes: deflection routing and fixed routing. The simulation was iplemented in RESQ (Research Queueing Package) [37] with the following assumptions:

- link delay= 1 unit of time, the time base,
- uniform and non-uniform traffic pattern. For the non-uniform case, a switch, say, $switch_i$, chooses each packet destination independently based on the following expression:
$$destination_i = Random_i\{1,16\} \cdot \phi + Fix_i\{1,16\} \cdot (1-\phi)$$
where, $Random_i$ returns an integer random value uniformly distributed between 1 and 16, excluding i. Fix_i returns a fixed integer value arbitrarily chosen at the beginning of the computation which excludes i. ϕ determines the percentage of traffic that is uniform.
- exponential interarrival time of new packets (local packets) with waiting room equal to one (packets that arrive while the buffer is in use are lost). The limit of one buffer is convinient because it limits the number of states of the simulation. This condition, however, also makes sense in a real system when hand-shake is used (i.e. a module can submit a new packet only after the previous one has been accepted by the network).

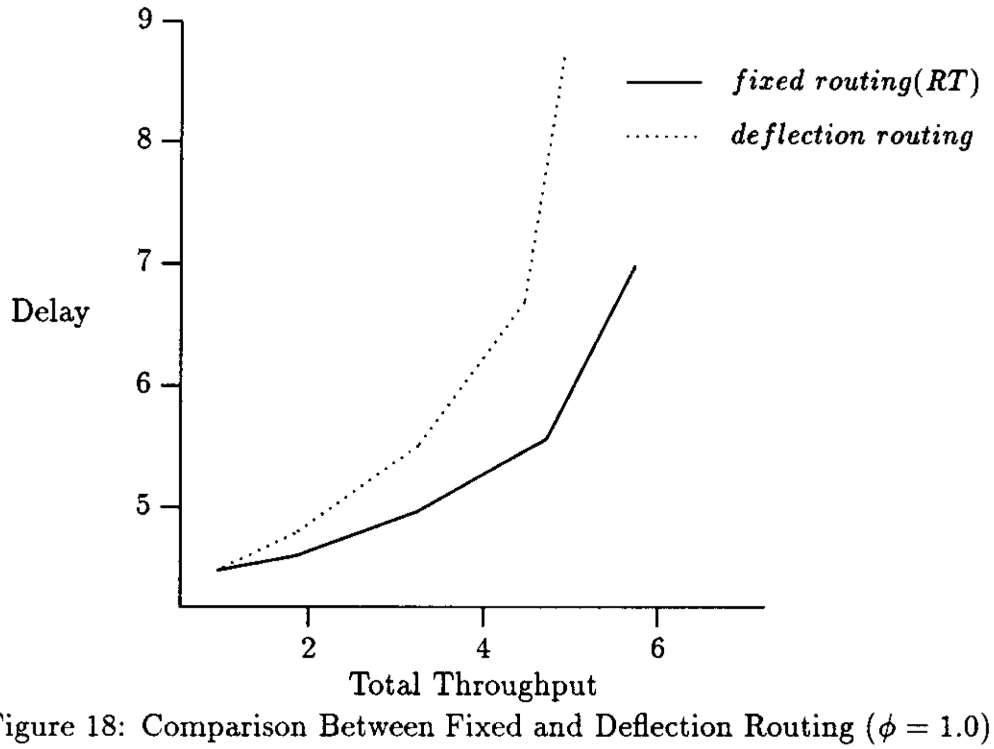

Figure 18: Comparison Between Fixed and Deflection Routing ($\phi = 1.0$)

The result for the comparison between fixed and deflection routing can be seen in Figure 18. As expected, deflection routing has higher delays than fixed routing because of longer paths the packets might take. When this happens, the packet will "stay" longer in the network, and occupy frames that otherwise could be used by other packets. Two other drawbacks of deflection routing are the fact that delay is not bounded and that packets might arrive out of order.

Figures 19 and 20 show the performance of the MRing network with fixed routing, under different load conditions and traffic patterns. As expected, performance is better in uniform load condition. We should also note that delay is bounded, since the waiting room for new packets is one and packets follow a fixed path to their destination.

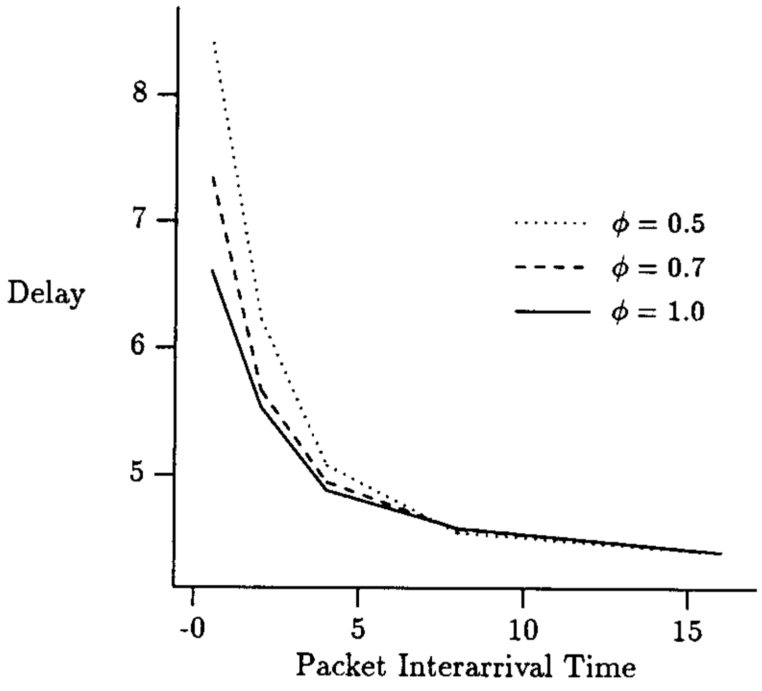

Figure 19: Delay versus Packet Interarrival Time

8 Concluding Remarks

The development of the VLSI technology has been very beneficial to the computer field by achieving greater integration and faster circuits. On the other hand, this development uncovers challenging problems. One of these problems is the long distance interconnection on chip which is increasingly becoming the dominant factor in the performance MOS technology integrated circuit, as well as in chip real estate (area consumption).

This work proposes new communication mechanisms that achieve cost effective solutions to overcome the high delay of signals over long wires. The problem of area usage is minimized by sharing the communication resources.

An important and interesting finding was that delay could be lowered and at the same time throughput could be increased on a long distance interconnection by segmenting the line and buffering the signal at these points. This implies that the preferred network solution

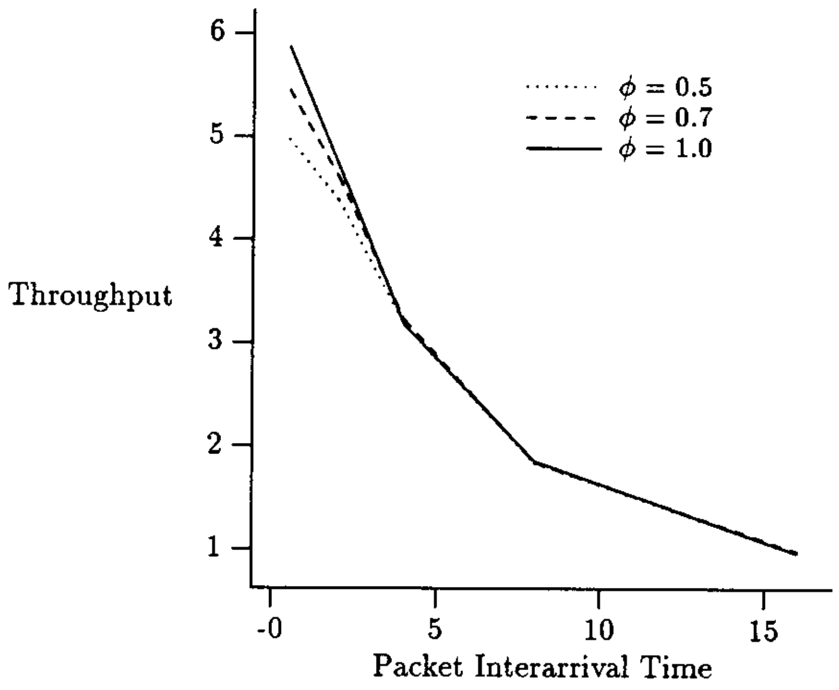

Figure 20: Throughput versus Packet Interarrival Time

for VLSI is slotted.

Furthermore, the idea of a network on chip is very attractive since it helps in dealing with the complexity of VLSI design by enforcing the hierarchical methodology approach. It also defines a common interface by which modules communicate. This makes easier the design effort of a complex system. Sharing of the resources on chip is likewise facilitated by the network as well as testing, maintenance, and fault isolation.

We also examine protocols for the proposed VLSI network discussing issues like liveness, flow control, fairness, and buffer needs. Performance results based on simulation were also discussed.

Appendix A Description of the DIRCP_BP

Algorithmically, the DIRC_BP access protocol can be described as:

```
initially, counter ← C, alt ← TRUE

at frame tick do
    if packet arrived then
        buffer it local_in buffer
        mark frame as free
    fi
```

```
        if counter $\neq 0$ and local packet waiting
                and no frame request from downstream then
            counter ← counter−1
        fi
        if frame requested then
            buffer incoming packet (if any)
            release free frame
        else if a through packet is present then
                (* either buffered or from the incoming frame *)
            send it
        else if local packet waiting then
            send it
            counter ← C
        fi
        if (counter = 0 and through_buffer > 0)
                or through_buffer > 1 then
            request frame from upstream node
            alt ← FALSE
        else if alt = FALSE then
            alt ← TRUE
        else if counter = 0 or through_buffer > 0 then
            request frame from upstream node
            alt ← FALSE
        fi
    od
```

Appendix B   Analytical Model for the Slotted Ring

The performance measurements that we are interested in for the slotted ring network using the DIRC_BP protocol are the time delay and throughput in a node for both, local and through traffic so that we may calculate the total time for a packet in the network as well as the total throughput of the network.

A $node_n$ (as seen in Figure 21) has three input flows, namely, the effective local packet flow (λ_{l_n}), the through packet flow coming from upstream $(\lambda_{t_{n-1}})$, and the request for free frames coming from downstream $(\lambda_{r_{n+1}})$. λ_{l_n} is the effective traffic that is accepted by $node_n$ when the "pressure" is λ_n, because nodes have only one buffer for local traffic. The solution for a N node ring is the set of values for $(\lambda_{l_n}, \lambda_{t_{n-1}}, \lambda_{r_{n+1}})_{1\leq n\leq N}$ for a given set of load requirement defined by $(\lambda_n)_{1\leq n\leq N}$ and by the packet destination address distribution.

The analytical model for this slotted ring architecture is based on a discrete Markov chain for the nodes. The transition probabilities between the states of the Markov chain

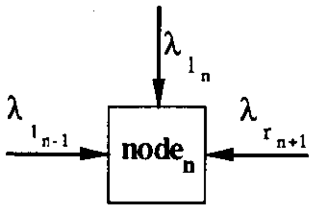

Figure 21: Input Flows to a Node

are functions of input flows to the nodes. The state space is composed of one absorbing state for the transmission of a local packet S_t, and states defined by $S_{i,b,c}$. The subscript i captures the condition that the node itself has requested a free frame, implying that the next frame coming to the node is a free frame (therefore, it cannot be a through packet); b counts the number of through packets buffered (which is bounded by two, see Section 5); and c keeps track of the value of the counter. Figure 22 illustrates the Markov chain for a node. Solving the Markov chain for a given set of input flows, we obtain the probability distribution for sending a local packet, an approximate distribution for sending a through packet, and the probability of requesting a free frame, from which we calculate the time delays and throughput.

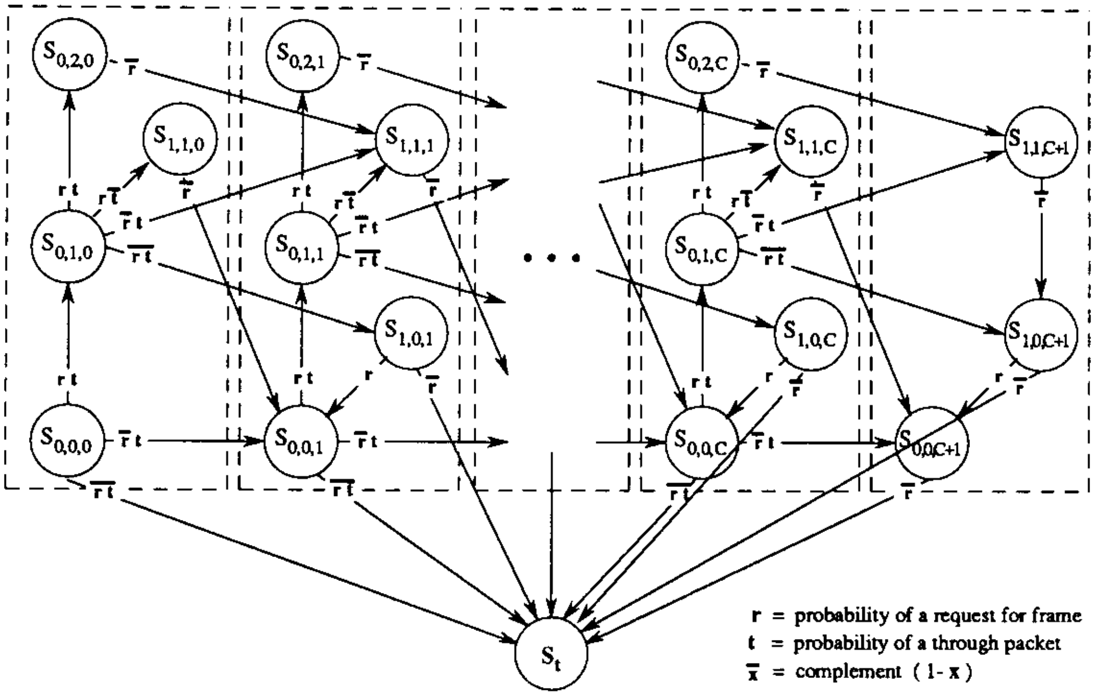

Figure 22: Markov Chain for a Node

In order to simplify the problem and solve it for the network, we assume that all packets are identically and independently distributed with uniform destination. With this, the state transition probabilities become node independent.

In our implementation, we solve the Markov chain using the power method. The initial state probability distribution is $S_{0,0,0} = 1$, with the remainder states equal to 0. The stopping condition is when the $Probability[S_t] - 1 < \epsilon,\ \epsilon \to 0$.

Since the output flows are monotonic functions for the input flows, we use an algorithm which is based on a fixed point solution. The result is shown in Figure 15, where it is compared with simulation results.

Appendix C Description of the Mring Access Protocol

The algorithmic description is:

```
initially, counter $\leftarrow$ C, alt $\leftarrow$ x, set alt_local $\leftarrow$ x
obs.: alt = x implies in $\overline{\text{alt}}$ = y and vice versa
    queue(&alt) means queue length of through traffic on the queue pointed by alt

at frame tick do
  if counter $\neq$ 0 then
    counter $\leftarrow$ counter $-$ 1
  if frame requested then
    release free frame
  else if alt = alt_local and counter = 0 and local packet ready then
    send local
    counter $\leftarrow$ C
    alt_local $\leftarrow$ $\overline{\text{alt\_local}}$
  else if queue(&alt) > 0 then
    send it
  else if queue(&$\overline{\text{alt}}$) > 0 then
    send it
  else if local packet ready then
    send local
    counter $\leftarrow$ C
    alt_local $\leftarrow$ $\overline{\text{alt\_local}}$
  if queue(&alt) > 1
    request free frame(alt)
  if queue(&$\overline{\text{alt}}$) > 2
    request free frame($\overline{\text{alt}}$)
  alt $\leftarrow$ $\overline{\text{alt}}$
od
```

References

[1] K. Saraswat and F. Mohammadi. Effect of scaling of interconnections on time delay of vlsi circuits. *IEEE Journal of Solid State Circuits*, SC-17(2):275–280, April 1982.

[2] C. P. Yuan and T. N. Trick. Calculation of capacitance in vlsi circuits, 1984.

[3] Charles Seitz. Concurrent vlsi architectures. *IEEE Transactions on Computers*, C-33(12):1247–1265, December 1984.

[4] Allan L. Fisher and H. T. Kung. Synchronizing large vlsi processor arrays. *IEEE Transactions on Computers*, c-34(8):734–740, August 1985.

[5] Francois Anceau. A synchronous approach for clocking vlsi systems. *IEEE Journal of Solid State Circuits*, SC-17(1):51–56, February 1982.

[6] Pinaki Mazumder. Evaluation of on-chip static interconnection networks. *IEEE Transactions on Computers*, C-36(3):365–369, March 1987.

[7] C. D. Thompson. *A Complexity Theory for VLSI.* PhD thesis, Carnegie-Mellon University, Pittsburg, PA, 1980.

[8] Stuart K. Tewksbury and L. A. Hornak. Communication network issues and high-density interconnects in large-scale distributed computing systems. *IEEE Journal on Selected Areas in Communications*, 6(3):587–609, April 1988.

[9] R. H. Dennard, F. H. Gaensslen, H. N. Yu, V. L. Rideout, E. Bassous, and A. R. LeBlanc. Desing of ion implanted mosfet with very small physical dimensions. *IEEE Journal of Solid State Circuits*, SC-9:256–268, October 1974.

[10] Carver Mead and Lynn A. Conway. *Introduction to VLSI Systems.* Addison-Wesley Publishing Company, October 1980.

[11] Neil Weste and Kamran Eshraghian. *Principles of CMOS VLSI Design - A Systems Perspective.* Addison-Wesley, October 1985.

[12] Donald S. Gardner, James D. Meindl, and Krishna C. Saraswat. Interconnection and electromigration scaling theory. *IEEE Transaction on Electron Devices*, ED-34(3):633–643, March 1987.

[13] Zhen-Qiu Ning, Patrick M. Dewilde, and Fred L. Neerhoff. Capacitance coefficients for vlsi multilevel metallization lines. *IEEE Transactions on Electron Devices*, ED-34:644–649, March 1987.

[14] W. Richard Smith, Scott Powell, and George Persky. A missing neighbor model for capacitive loading in vlsi interconnect channels. *IEEE Journal of Solid-State Circuits*, SC-22(4):553–557, August 1987.

[15] H. B. Bakoglu and James D. Meindl. Optimal interconnection circuits for vlsi. *IEEE Transaction on Electron Devices*, ED-32(5):903–909, May 1985.

[16] Don Speck. Mos vlsi trendlines. Usenet distribution, September, 16 1988.

[17] T. Sakurai and K. Tamaru. Simple formulas for two- and three-dimensional capacitances. *IEEE Trasanctions on Electron Devices*, ED-30(2):183–185, February 1983.

[18] Lance A. Glasser and Daniel W. Dobberpuhl. *The Design and Analysis of VLSI Circuits.* Addison-Wesley, 1985.

[19] Amar Mukherjee. *Introduction to nMOS and CMOS VLSI Systems Design.* Prentice-Hall, Englewood Cliffs, NJ, 1985.

[20] Nils Hedenstierna and Kjell O. Jeppson. Cmos circuit speed and buffer optimization. *IEEE Transaction on Computer Aided Design*, CAD-6(2):270–281, March 1987.

[21] H. C. Card, W. Pries, and R. D. McLeod. Contributions to vlsi computational complexity theory from bounds on current density. *The VLSI Integration*, 4:175–183, 1986.

[22] Vijaya Ramachandran. On driving many long lines in a vlsi layout. In *Proceedings 23rd Symposium on the Foundations of Computer Science*, pages 369–378, 1982.

[23] Carver Mead and Martin Rem. Minimum propagation delays in vlsi. *IEEE Journal of Solid State Circuits*, SC-17(4):773–775, August 1982.

[24] Frank Schaffa. *Communications on VLSI.* PhD thesis, Computer Science Department, University of California, Los Angeles, CA 90024, 1989.

[25] Arnold Reisman. Device, circuit, and technology scaling to micron and submicron dimensions. *Proceedings of the IEEE*, 71(5):550–565, May 1983.

[26] H. Bakoglu and J. Meindl. Optimal interconnect circuits for vlsi. In *Proceedings 1984 IEEE International Solid State Circuits*, pages 164–165, February 1984.

[27] H. Richard Gail. *On the Optimization of Computer Network Power.* PhD thesis, Computer Science Department, University of California, Los Angeles, CA 90024, 1983.

[28] H.T. Kung. Why systolic architectures? *IEEE Computer Magazine*, 15(1):37–46, January 1982.

[29] Charles H. Stapper, Frederick M. Armstrong, and Kiyotaka Saji. Integrated circuit yield statistics. *Proceedings of the IEEE*, 71(4):453–470, April 83.

[30] Israel Koren and Melvin A. Breuer. On area and yield considerations for fault tolerant vlsi processor arrays. *IEEE Transaction on Computers*, C-33(1):21–27, January 1984.

[31] Michael Fine and Fouad A. Tobagi. Demand assignment multiple access schemes in broadcast bus local area networks. *IEEE Transactions on Computers*, c-33(12):1130–1159, December 1984.

[32] R. M. Fujimoto. *VLSI Communication Components for Multicomputer Networks.* PhD thesis, Computer Science Division (EECS), University of California, Berkeley, CA 94720, 1983. Published as Technical Report UCB/CSD 83/136.

[33] J. R. Pierce. Network for block switching of data. *Bell System Technical Journal*, 51:1133–1143, July/August 1982.

[34] N. F. Maxemchuk. Regular mesh topologies in local and metropolitan area networks. *AT&T Technical Journal*, pages 1659–1685, September 1985.

[35] N. F. Maxemchuk. Routing in the manhattan street network. *IEEE Trasactions on Communications*, COM-35(5):503–512, May 1987.

[36] Albert G. Greenberg and Jonathan Goodman. Sharp approximate models of adaptive routing in mesh networks. *Teletraffic Analysis and Computer Performance Evaluation*, pages 255–270, 1986.

[37] Charles H. Sauer, Alvin Blum, Paul Loewner, Edward MacNair, and James Kurose. *The Research Queueing Package Version 2.* IBM, 1986.