A High-Performance Optoelectronic Interconnect Router: Using Increased Bandwidth to Enable Latency Reduction

Timothy M. Pinkston     Mongkol Raksapatcharawong     Yungho Choi
The SMART Interconnects Group
EE-Systems Dept., University of Southern California
Los Angeles, CA 90089-2562
{http://www.usc.edu/dept/ceng/pinkston/SMART.html}

Abstract:

Continuing strides in processor performance impose the need for aggressive latency hiding/reduction techniques in order to fully exploit improved capabilities at the system level. These techniques demand increased interconnection bandwidth which, ultimately, can surpass that which can be supplied by current technology. This paper addresses this problem by presenting the design and implementation of an advanced optoelectronic interconnection network router capable of supplying and efficiently utilizing higher bandwidth capacity to better enable various latency hiding/reduction techniques.

Keywords: deadlock, fully adaptive routing, interconnection network, latency hiding/reduction techniques, optoelectronic router.

1. Introduction

Since the introduction of the microprocessor about 25 years ago, microprocessor performance has doubled approximately every 18 months. This sustained improvement stems primarily from advances made at the technology level as well as at the architecture level. These advances enable processors to run at faster clock rates and execute many more instructions simultaneously. Continuing strides in processor performance impose the need for making communication latency more transparent to the critical path of computation, particularly for multiprocessor configurations. On-chip memory accesses through registers and various levels of cache can satisfy many instruction and data fetches, but some accesses must go off-chip through the memory hierarchy. This memory hierarchy spans several levels in multiprocessors, many of which require the use of a communication ether or interconnection network.

Interconnects traditionally have not achieved comparable strides in performance as processors, resulting in a widening latency gap between intraprocessor and interprocessor communication. With remote access latencies presently ranging from tens to hundreds of processor clock cycles and growing, multiprocessor systems must employ latency hiding, tolerating, and/or reduction techniques to offset this worsening problem. However, to be maximally effective, these techniques impose a stringent bandwidth demand on the interconnection network. High-performance interconnection networks that are capable of supplying and efficiently utilizing high I/O bandwidth are therefore needed to better enable techniques designed to mitigate the latency problem.

For this reason, emerging optoelectronic technology is becoming of increasing interest to computer architects owing to its potential for dense, high-speed implementation of high-bandwidth interconnects. Rapid progress in optoelectronic fabrication and packaging technology is making this alternative a nearer-term interconnect solution. These technologies include hybrid integration of CMOS VLSI circuitry with GaAs-based multiple quantum well modulators (via flip-chip bonding) with more than 10,000 I/Os per square centimeter, each operating beyond 2Gb/s using only 300uW of optical power in differential (dual-rail) signal mode [1]. This density of interconnection is more than quadruple the I/O capacity of widely used ball grid array (BGA) technology. Other emerging optoelectronic technologies include monolithic integration of GaAs MESFET logic circuitry with VCSEL/MSM and LED/OPFET sources and photodetectors that operate at greater than 9GHz [2,3], and hybrid integration of VCSELs with CMOS logic circuitry each operating at bandwidths greater than 2GHz [4].
In this paper, we present the design and implementation of an advanced optoelectronic network router capable of supplying and efficiently utilizing high bandwidth capacity. In the next section, we give background on the importance of high bandwidth capacity networks and efficient utilization of the bandwidth. In Section 3, we present advanced router architecture techniques based on fully-adaptive deadlock recovery routing for efficiently utilizing the high-bandwidth capacity of optoelectronic interconnects. In Section 4, we present the design and implementation of a high-performance optoelectronic interconnection network router that uses the technologies and techniques described previously and give preliminary performance results. Finally, we give conclusions in Section 5.

Examples of optoelectronic devices

**MSM** (Metal-Semiconductor-Metal) is a photodetector device comprised of metal electrodes deposited on top of a semiconductor material. When light with energy larger than the semiconductor band gap is incident, electron-hole pairs are generated and are quickly swept to the electrodes, resulting in current flow. This device can be very fast but does not exhibit gain.

**OPFET** is a GaAs-based photodetector device. It is an Enhancement-FET (EFET) with optical power input converted to an equivalent gate bias; its source input is connected to a diode-connected Depletion-FET (DFET) load to make an optical-in/DCFL-out inverter. It is slower than MSM photodetector but has higher output gain.

**SEEDs** (Self Electrooptic-Effect Devices) are AlGaAs-based passive diode devices invented at Lucent technologies (formerly Bell Labs). The devices change their light absorption when an applied electric field across the device terminals is varied. This results in light modulation controlled by an electrical signal to be used as a transmitter. The same device also works as a receiver that generates a leakage current upon receiving light, thus, converting an optical signal back to an electrical signal.

**VCSEL** (Vertical-Cavity Surface-Emitting Laser) is an active device that has a $p-n-p-n$ structure with an active layer and cavity mirrors at the top and bottom surfaces. The device structure is a stack of p-typed layers, undoped barrier layers, active layers, and n-typed layers. Some features expected are an ultra low threshold current, a very small size, high operational speed, and high-intensity light output.

2. Importance of High Network Bandwidth

Remote memory access time is decreasing at a slower rate than processor cycle time, creating a latency gap which can degrade overall system performance. The severity of this widening latency gap is shown in Table 1 which projects on-chip and off-chip cycle times and bus widths for memory and microprocessor technologies a decade into the future. Also, off-chip bandwidth demand (sustained) for current and future generation processors trending is juxtaposed with bandwidth supplied by current and future generation buses trending in Figure 1. It is evident that if the network provides insufficient bandwidth, the latency increases accordingly.

<table>
<thead>
<tr>
<th>Year</th>
<th>Smallest feature, $\mu$m</th>
<th>DRAM (High-Performance)</th>
<th>Microprocessor (High-performance)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Density, Mbits</td>
<td>On-chip/off-chip clock, MHz</td>
</tr>
<tr>
<td>1995</td>
<td>0.35</td>
<td>64</td>
<td>150/50</td>
</tr>
<tr>
<td>1998</td>
<td>0.25</td>
<td>256</td>
<td>200/66</td>
</tr>
<tr>
<td>2001</td>
<td>0.18</td>
<td>1024</td>
<td>300/100</td>
</tr>
<tr>
<td>2004</td>
<td>0.13</td>
<td>4096</td>
<td>400/100</td>
</tr>
<tr>
<td>2007</td>
<td>0.10</td>
<td>16384</td>
<td>500/125</td>
</tr>
<tr>
<td>2010</td>
<td>0.07</td>
<td>65536</td>
<td>625/150</td>
</tr>
</tbody>
</table>
There are several architectural techniques beyond caching that are particularly useful for dealing with the latency problem in multiprocessor systems. Prefetching [5] is a technique that fetches data in advance before it is actually required. Once the actual access (i.e. Load) occurs, data is already available or is on its way to the processor, therefore hiding latency. Multithreading [6] is another technique for hiding/tolerating latency. On a long latency operation, the processor switches to a new thread that is ready to run. Thus, on a pending access, the processor engages in useful work overlapping communication with computation. While both techniques can be used aggressively to hide/tolerate latency, both increase network traffic: prefetching can increase cache miss traffic by evicting useful blocks from the cache and can increase overhead traffic by fetching unused blocks; multithreading can increase cache capacity miss traffic by effectively reducing the cache size with the working sets of multiple threads and can increase sustained traffic by allowing multiple outstanding accesses to the network. For example, Figure 2 shows the increase in network bandwidth needed to enable multithreading for complete overlap of communication and computation. As can be seen, for maximum effectiveness, these techniques require a high bandwidth interconnection network.

A high bandwidth interconnect is necessary but not sufficient to aggressively mitigate the latency problem. Additional architectural techniques which allow efficient use of scarce bandwidth are also needed to reduce contention by competing processors. For example, off-chip bandwidth for next generation processors is projected to be about 6GB/s. This number could reach 60GB/s in the next decade. With multiple processors each demanding a considerable fraction of this from the network, even a high-bandwidth optoelectronic interconnection network could easily become oversaturated unless techniques to efficiently use the bandwidth are employed.
3. Efficient, Deadlock-free Utilization of High-Bandwidth Optoelectronic Interconnects

Advanced architectural techniques which exploit the connectivity advantages of optoelectronic interconnect technology are being explored in the design of multiprocessors in order to achieve the highest possible latency and throughput performance [7,8,9]. An essential component having substantial impact on the performance of the interconnection network is the network router. Routers are switching components that establish the topology of the network as well as determine the path packets take from source to destination. To efficiently utilize the increased optoelectronic bandwidth capacity and to increase network throughput, various architectural techniques can be used to enhance router performance. Among such techniques are wormhole switching, adaptive routing, and virtual channel flow control. Wormhole switching [10] is a technique that reduces the message latency by splitting a message equally into several units called "flits" and pipelining flit transfer. Adaptive routing [11] improves performance by allowing packets to freely propagate through one of several available paths, thereby evenly distributing traffic throughout the network as well as having the capability to avoid congestion and faults in the network. Virtual channel flow control [12] increases the utilization of physical links by time-multiplexing the transmission of multiple packets over the channel to improve packet flow and reduce message blocking.

The potential impact on network performance of optoelectronic technology integrated with these architectural techniques can be seen using a fundamental equation which represents the average packet latency of wormhole k-ary n-cube networks1 (Figure 3 shows a 4-ary 2-cube torus network):

\[
T_{\text{latency}} = T_C \left[ n \cdot \left( \frac{k - 1}{2} \right) + \frac{L}{W} \right] + T_{\text{contention}},
\]

Here \(T_{\text{latency}}\) is the average packet latency, \(T_C\) is the channel cycle time (or network clock period), \(n\) is the dimension of the network, \(N\) is the number of nodes in the system, \(k\) is the radix of the network, \(L\) is the packet size, \(W\) is the channel width, and \(T_{\text{contention}}\) is the time packet wait to use the channels due to contention. Simply, the first term in the bracket represents the number of hops between source and destination, and the second term represents the number of flits to be transmitted. Wormhole switching makes the two terms additive rather than multiplicative to reduce latency.

Channel cycle time benefits from a distance-independent, low latency interconnect technology such as optoelectronics. This becomes more important as the router logic becomes faster and, hence, the network cycle becomes determined by external router delay. The higher I/O density provided by optoelectronic technology also has a direct impact on reducing both the number of hops packets experience and the number of flits that must be transmitted because of the potential to increase node degree and channel width. Higher node degree implies high dimensional networks that are believed to be less efficient (due to wiring complexity) than low dimensional networks in electrical interconnect technology. However, a study of optically interconnected networks [13] shows that this is not the case with optoelectronic implementations.

Adaptive routing and virtual channel flow control are very important in reducing contention time (latency) by increasing the availability of network bandwidth to packets. The most efficient utilization of network bandwidth can be achieved by allowing true fully adaptive routing in which unrestricted routing of packets towards their destination on all virtual channels in all network dimensions is allowed. Although deadlocks must be guarded against, the higher degree of routing freedom provided gives packets a higher chance of circumventing congestion and utilizing network bandwidth more efficiently to reduce latency. To illustrate this, Figure 3 shows a two-dimensional torus network with three virtual channels per physical link. The routing freedom allowed by three routing algorithms are illustrated: the nonadaptive (static) dimension order routing algorithm which routes packets in dimension order to avoid deadlock (i.e., used in the SGI SPIDER and Cray T3D routers), the adaptive Duato Protocol routing algorithm which routes packets in a less restricted manner but still enforces some routing restrictions to avoid deadlock (i.e., used in the MIT Reliable Router and Cray T3E router), and the Disha true fully-adaptive routing algorithm which allows complete unrestricted routing (i.e., used in the WARRP router).

---

1 A network that belongs to the k-ary n-cube class of networks has n dimensions with k nodes in each dimension for a total number of nodes of \(k^n\).
Dimension order routing forces packets to traverse dimensions in some fixed order (e.g., X then Y), providing packets with at most only two virtual channel options at each intermediate router or a total of 16 different paths from source to destination as shown in Figure 3(a). Duato’s Protocol allows one additional virtual channel to be used corresponding to the adaptive channel in the nondimension-ordered direction for a total of 36 different paths as shown in Figure 3(b). If, however, these permissible channels all happen to be occupied by other packets enroute, this packet must remain blocked even if the other (restricted) channels are free. True fully adaptive routing, on the other hand, allows packets to access all virtual channels along any path towards the destination (only minimal ones are shown here) for as many as six virtual channel options at each intermediate router or a total of 486 different routing paths, as shown in Figure 3(c). As shown in Figure 4 and confirmed in previous work [14], this greater routing freedom allows for significant improvement in network performance (latency and throughput) by increasing the utilization of available network bandwidth.

![Figure 3. Routing adaptivity comparison.](image)

![Figure 4. Latency and throughput comparison for various routing techniques.](image)
How do deadlocks form?

Allowing unrestricted routing can result in packets forming resource wait-for cycles [15], some of which can lead to deadlock. Deadlock arises from the endless blocking of packets waiting to acquire network resources held by others. Impending deadlocks must be efficiently recovered from to resume normal packet routing. An example of deadlock is depicted in Figure 5. Here, Packet $m_1$ is holding routing channels $vc_0$, $vc_1$, and $vc_2$ while requesting another channel $vc_5$; Packet $m_2$ is similarly holding routing channels $vc_3$, $vc_4$, and $vc_5$ while requesting another channel $vc_8$, etc. Packets $m_1$, $m_2$, $m_3$, and $m_4$, form a circular-waiting resource dependency that leads to deadlock.

Although deadlocks are possible with true fully-adaptive routing, previous work [16,17] has shown that they are highly improbable when sufficient routing freedom is provided by the interconnection network and fully exploited by the routing algorithm. As the number of network resources (i.e., physical and virtual channels) and the routing options allowed on them (i.e., true fully adaptive routing) increase, routing freedom increases (see Figure 6). As routing freedom increases, the number of messages which tend to block decreases which decreases the number of resource dependency cycles and the probability of deadlock exponentially. This is because multicycle deadlocks require highly correlated patterns of dependency cycles, the complexity of which increases with routing freedom. Hence, there is growing interest in deadlock recovery-based routers which allow true fully-adaptive routing.

![Figure 5. A simple resource wait-for cycle that leads to deadlock.](image1)

![Figure 6. Adding virtual channel increases routing freedom.](image2)
4. A High-Bandwidth Optoelectronic Interconnect Router

Below, we describe the design of an efficient, true-fully-adaptive interconnection network router called WARRP and its optoelectronic implementation (WARRP II).

4.1 Overview of the WARRP Router

The WARRP router (Wormhole Adaptive Recovery-based Routing via Preemption) implements a progressive deadlock recovery-based routing algorithm which allows unrestricted true-fully-adaptive wormhole routing of packets for maximum utilization of network bandwidth. Figure 7 shows a block diagram of the router which is designed for two-dimensional torus topologies. It provides high-bandwidth communication through four bidirectional physical links directly connecting neighboring routers plus one bidirectional injection/reception link connecting its local processor node. Packets of arbitrary length can be routed but are divided into 8-bit flits. Each physical link implements three bidirectional virtual lanes. The internal datapath of the router is equal to the external link width.

The router is designed with a bipartite structure that separates deadlock handling resources from normal (non-deadlock) routing resources so that each can be optimized separately to achieve maximum efficiency. The internal router architecture is further optimized by having a partitioned crossbar structure organized by virtual channel network [18]. This allows the WARRP router to implement the highest degree of adaptivity while not sacrificing router speed, regardless of the number of virtual channels per physical link. It also allows for unrestricted routing using normal routing resources while still ensuring deadlock freedom.

Deadlock freedom is guaranteed by redirecting potentially deadlocked packets to flit-sized deadlock buffers central to the routers which preempt physical channel bandwidth in order to synthesize recovery. Access to deadlock buffers is controlled (i.e., access granted by capturing a circulating token) so that, system-wide, these deadlock handling resources form what is essentially a deadlock-free lane on which potentially deadlocked packets can be delivered to resolve impending deadlock. On the event of an impending deadlock, an eligible packet which captures the token is switched to the deadlock-free lane starting at the current router's deadlock buffer and routed minimal adaptively along a recovery path leading to its destination where it is consumed to break the impending deadlock. Network bandwidth may be temporarily deallocated from other “normal” packets and assigned to the “deadlock” packet during the recovery process and re-allocated back to preempted normal packets when the deadlock packet's tail passes. Hence, network bandwidth is dynamically allocated to recover from deadlock only in the rare instances when impending deadlock is suspected; otherwise, all network bandwidth is preallocated to true fully adaptive routing of normal packets.
The internal structure of the normal routing section is partitioned into three virtual network modules as shown in Figure 8. Incoming packets are directed to one of the virtual networks in the normal routing
section or to the **deadlock module** in the deadlock handling section, depending on the status of the control signals sent along with the packet (signifying deadlock or normal). A normal packet entering the router is received by the **input module** of the corresponding virtual network module. Routing information is extracted from the **header flit** (first flit of the packet) by the **address decoder** which sequentially makes requests to the **routing and arbitration module** for a permissible output link by cycling through the following request types until being granted: (1) an **output module** in the current virtual network, (2) an **output module** in the other dimension of the same virtual network, (3) a **connect channel** to the next virtual network, and (4) the recovery lane in the deadlock module (only eligible packets are allowed to make this request). If the first request type is granted, the input-buffered packet is routed over the current virtual network’s **crossbar** to the allocated output module after the header is updated. If a connect channel is allocated to the packet due to all the available output links in the current virtual network being occupied by other packets, the packet is transferred to the next virtual network’s input module where the same routing actions are performed. If the recovery lane request is granted due to the permissible output modules and connect channel of the virtual network being busy, the crossbar is configured to redirect the suspected deadlock packet to the deadlock module to invoke the recovery process once the token is captured. The status of the packet is then changed from "normal" to "deadlock". As a final step, packets are time-multiplexed onto the physical link from the output modules of the three virtual networks in a round-robin fashion or given priority over the link if coming from the deadlock module. The input, output, and deadlock modules are deallocated from the packet once the **tail flit** passes.

A deadlock packet entering the router is latched directly into the central deadlock module in the deadlock handling section of the router. The header flit of the buffered deadlock packet is decoded by the address decoder of the deadlock module. This routing information is used to decide which physical link will be allocated to the deadlock packet, preempting other normal packets which may be routing over the link. The control of normal and deadlock packet flow between routers is regulated by the **flow control** module to ensure no overflow. The **WARRP** router implements these functions using the major components described further in [19].

### 4.2 Optoelectronic Implementation of the WARRP Router: WARRP II

The WARRP II router implementation addresses the pin-out problem by integrating dense, high-bandwidth optoelectronic transceivers onto a scaled-down version of the WARRP router design. This technology is capable of providing over 300GB/s of optical I/O per square centimeter [20]. The WARRP II router is being implemented on a 2x2mm² chip using a 0.5μm (with 0.7μm drawn gate length), 3-metal layer, 3.3V supply voltage CMOS process. The WARRP II layout, shown in Figure 9, contains approximately 15,000 transistors in total, of which 3,000 transistors are used for transceiver and pad driver circuits. It is a fully functional network router which incorporates one of the WARRP virtual networks and the deadlock recovery mechanism, representing a 4-bit wide unidirectional torus network router. These components (shown in a boxed area in Figure 8) are sufficient to fully demonstrate the deadlock formation, detection, and recovery operations.

Self Electrooptic-Effect Devices (SEEDs) [21] are integrated onto CMOS-VLSI circuitry of the WARRP II to enable optical interconnects among chips. These devices are flip-chip bonded on top of the CMOS circuitry, organized as an array of 20x10 diodes. A maximum of 100 dual-rail signals can be represented on a 2x2mm² chip. Each diode is 20x60μm² and is separated by 62.5μm and 125μm from its neighbors in X- and Y-directions, respectively.

The WARRP II router is packaged in an 84-pin PGA but uses only 18 optical (dual-rail) and 23 electrical signals, as output and input signals. We have simulated the WARRP II under both IRSIM (shown in Figure 10) and HSPICE. The simulation results indicate that it can achieve at least 30 MHz clock rate, limited by the datapath logic circuitry not optical I/O. Under no contention, it can route packets in 2 cycles and yield a throughput of one flit per cycle, reaching 30MB/s aggregate bandwidth.
5. Conclusion

Latency has long time been recognized as a problem in computer systems, both in uniprocessor and multiprocessor systems. This problem is being exacerbated by the technology trends in processor speed and bandwidth requirement, making more off-chip bandwidth and latency hiding mechanisms necessary. Increasing interconnection bandwidth alone is not sufficient to mitigate the latency problem. Aggressive techniques to efficiently utilize scarce bandwidth are also needed.

With the emergence of optoelectronic technology, we are able to design high-bandwidth network routers that support architectural techniques that efficiently utilize the increased bandwidth. In this paper we have presented the design and implementation of the WARRP II multiprocessor router chip that takes advantage of high-bandwidth optical input/output and incorporates advanced architectural techniques such as true fully adaptive routing, progressive deadlock recovery, and enhanced internal router design. We believe the WARRP II design will lead to the continuation of optical interconnection network designs that support bandwidth-intensive architectural techniques to deal effectively with the latency problem.
6. References


