Improving Energy Efficiency of Network-on-Chips Using Emerging Wireless Technology and Router Optimizations

A thesis presented to
the faculty of
the Russ College of Engineering and Technology of Ohio University

In partial fulfillment
of the requirements for the degree
Master of Science

Dominic F. DiTomaso

June 2012

© 2012 Dominic F. DiTomaso. All Rights Reserved.
This thesis titled
Improving Energy Efficiency of Network-on-Chips Using Emerging Wireless Technology
and Router Optimizations

by
DOMINIC F. DITOMASO

has been approved for
the School of Electrical Engineering and Computer Science
and the Russ College of Engineering and Technology by

___________________________________________
Avinash Kodi
Assistant Professor of Electrical Engineering and Computer Science

___________________________________________
Dennis Irwin
Dean of Russ College of Engineering and Technology
ABSTRACT

DITOMASO, DOMINIC F., M.S., June 2012, Electrical Engineering

Improving Energy Efficiency of Network-on-Chips Using Emerging Wireless Technology and Router Optimizations (84 pp.)

Director of Thesis: Avinash Kodi

As silicon technology scales down to the sub-nanometer region, chip designs are moving toward chip multiprocessors (CMPs), integrating hundreds and even thousands of smaller cores on a single chip. As such, the design and implementation of the underlying communication fabric is becoming a critical challenge. Traditional bus-based networks have problems with scalability and latency. Network-on-Chip (NoC) architectures overcome these problems by using packet switched networks that connect cores in topologies such as meshes or tori. However, traditional NoC topologies use metallic interconnects which cause high latency and power consumption as technology nodes scale down in size. Additionally, the routers in packet switched networks have power hungry buffers and large crossbars. A potential solution to the interconnect problems is wireless links. Wireless interconnects can provide low power and low latency communication between cores. In this thesis, a wireless architecture called iWISE is proposed, which is a scalable, wireless/wired hybrid network that reduces power and improves throughput and latency over other wired and wireless networks. To overcome the router buffer and crossbar problems, a CTorus topology is proposed in a separate network which uses low power channel buffers and smaller, split crossbars to improve overall network performance and power.

Approved: ________________________________

Avinash Kodi

Assistant Professor of Electrical Engineering and Computer Science
I dedicate this thesis to my parents, sister, and grandparents for their love and support.
# Table of Contents

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
</tr>
<tr>
<td>Dedication</td>
</tr>
<tr>
<td>List of Tables</td>
</tr>
<tr>
<td>List of Figures</td>
</tr>
<tr>
<td>1 Introduction</td>
</tr>
<tr>
<td>1.1 NoC</td>
</tr>
<tr>
<td>1.1.1 NoC Router Microarchitecture</td>
</tr>
<tr>
<td>1.1.2 NoC Pipelining and Flow Control</td>
</tr>
<tr>
<td>1.1.3 NoC Topologies</td>
</tr>
<tr>
<td>1.2 Challenges with Traditional NoCs</td>
</tr>
<tr>
<td>1.3 Wireless/RF Technologies</td>
</tr>
<tr>
<td>1.4 Router Optimizations</td>
</tr>
<tr>
<td>1.5 Major Contributions</td>
</tr>
<tr>
<td>1.6 Organization of Thesis</td>
</tr>
<tr>
<td>2 Emerging Technology: Wireless Interconnects</td>
</tr>
<tr>
<td>2.1 Architecture: iWISE-64</td>
</tr>
<tr>
<td>2.2 Architecture: iWISE-256</td>
</tr>
<tr>
<td>2.3 Wireless Technology</td>
</tr>
<tr>
<td>2.4 Challenges with Wireless Technology</td>
</tr>
<tr>
<td>3 Router Optimizations</td>
</tr>
<tr>
<td>3.1 Channel Buffers</td>
</tr>
<tr>
<td>3.1.1 Dual Channel Organization</td>
</tr>
<tr>
<td>3.1.2 Dual Channel Multi-Input Organization</td>
</tr>
<tr>
<td>3.1.3 Single Channel Multi-Input Organization</td>
</tr>
<tr>
<td>3.2 Crossbars</td>
</tr>
<tr>
<td>3.2.1 Dual Crossbar Organization</td>
</tr>
<tr>
<td>3.2.2 Multi Crossbar Organization</td>
</tr>
<tr>
<td>3.3 CTorus</td>
</tr>
<tr>
<td>4 Results and Discussion</td>
</tr>
<tr>
<td>4.1 Wireless Technologies</td>
</tr>
<tr>
<td>4.1.1 Performance</td>
</tr>
<tr>
<td>4.1.2 Power</td>
</tr>
</tbody>
</table>
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>Core and cache parameters used while extracting traces from SIMICS with GEMS [1]</td>
<td>52</td>
</tr>
<tr>
<td>4.2</td>
<td>Power estimation per mm link using Synopsys Design Compiler with the 65 nm TSMC technology libraries at 1.0 V and 1 Ghz clock [2].</td>
<td>65</td>
</tr>
<tr>
<td>4.3</td>
<td>Area estimation for buffers with 1 mm links and crossbars using Synopsys Design Compiler with the 65 nm TSMC technology libraries [2].</td>
<td>70</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Trend of CMPs due to the steady increase of transistors, leveling off of single core performance, clock frequency, and power [3].</td>
</tr>
<tr>
<td>1.2</td>
<td>Traditional router microarchitecture showing buffers, crossbar, route computation (RC), switch allocator (SA), and virtual channel allocator (VA).</td>
</tr>
<tr>
<td>1.3</td>
<td>(a) Baseline pipelining stages, (b) baseline using lookahead routing, and (c) baseline using look-ahead routing and SA speculation</td>
</tr>
<tr>
<td>1.4</td>
<td>Leading NoC topologies (a) Mesh, (b) Concentrated Mesh (CMesh), and (c) Flattened Butterfly (FBfly).</td>
</tr>
<tr>
<td>1.5</td>
<td>Related work: (a) Hybrid Subnets [4], (b) WCube [5], (c) RF-Interconnect [6], and (d) Irregular Wireless [7].</td>
</tr>
<tr>
<td>2.1</td>
<td>iWISE architecture showing the communication set for one cluster [1].</td>
</tr>
<tr>
<td>2.2</td>
<td>(a) iWISE-64 using the token-partial communication where ( C = 4 ) and ( S = 4 ) and (b) communication example from set 0 to set 1 [1].</td>
</tr>
<tr>
<td>2.3</td>
<td>Router microarchitecture for iWISE-64.</td>
</tr>
<tr>
<td>2.4</td>
<td>Scalable architecture for iWISE-256 showing the wireless communication between sets where ( C = 4, S = 4, ) and ( G = 4 ) [1].</td>
</tr>
<tr>
<td>2.5</td>
<td>Partial view of iWISE-256 with an inter-group and intra-group communication example [1].</td>
</tr>
<tr>
<td>2.6</td>
<td>Router microarchitecture of iWISE-256 showing set router and group router [1].</td>
</tr>
<tr>
<td>2.7</td>
<td>Transceiver design implemented in 32 nm RF-CMOS centered around 100 GHz with OOK modulation operating at 5 Gbps for 16 channels [8].</td>
</tr>
<tr>
<td>3.1</td>
<td>(a) Modified tri-state repeaters (shown in inset) capable of storing, (b) control logic, and (c) state transition diagram [2].</td>
</tr>
<tr>
<td>3.2</td>
<td>Tri-state inverters replacing conventional repeaters in (a) a dual channel (dc) buffer organization, (b) a dual channel multi-input (dcM) organization, and (c) a single channel multi-input (scM) organization [2].</td>
</tr>
<tr>
<td>3.3</td>
<td>(a) Baseline crossbar organization, (b) dual crossbar (dx) organization, and (c) multi crossbar (mx) organization [2].</td>
</tr>
<tr>
<td>3.4</td>
<td>CTorus topology using dc buffer organization and mx crossbar organization [9].</td>
</tr>
<tr>
<td>4.1</td>
<td>Speedup simulations relative to mesh for 64 core topologies under SPLASH-2, PARSEC, and SPEC CPU2006 benchmarks. blackscholes (b-scholes), fluidanimate (f-animate), streamclusters (s-clusters), and swaps (swaptions) [1].</td>
</tr>
<tr>
<td>4.2</td>
<td>64 core saturation throughput for synthetic traffic patterns: Uniform (UN), Non-Uniform Random (NUR), Bit Reversal (BR), Butterfly (BFLY), Complement (COMP), Matrix Transpose (MT), Perfect Shuffle (PS), Neighbor (NBR), and Tornado (TN).</td>
</tr>
<tr>
<td>Section</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>4.3</td>
<td>(a)-(c) Average network latency per packet and (d)-(f) throughput</td>
</tr>
<tr>
<td>4.4</td>
<td>256 core saturation throughput for synthetic traffic patterns: Uniform (UN), Non-Uniform Random (NUR), Bit Reversal (BR), Butterfly (BFLY), Complement (COMP), Matrix Transpose (MT), Perfect Shuffle (PS), Neighbor (NBR), and Tornado (TN) [1].</td>
</tr>
<tr>
<td>4.5</td>
<td>Average power per packet of 64 core topologies under synthetic traffic loads</td>
</tr>
<tr>
<td>4.6</td>
<td>Average power per packet of 256 core topologies under synthetic traffic loads [1].</td>
</tr>
<tr>
<td>4.7</td>
<td>Area overhead of routers, wired, and wireless links for (a) 64 core networks and (b) 256 core networks.</td>
</tr>
<tr>
<td>4.8</td>
<td>Baseline and dc-mx organization power consumption as technology changes from 65 nm to 40 nm [10].</td>
</tr>
<tr>
<td>4.9</td>
<td>Power per packet for different traffic loads. (a) Complement (b) Butterfly, and (c) Uniform Random [9].</td>
</tr>
<tr>
<td>4.10</td>
<td>Relative Energy-Delay Product (EDP) of CTorus and other 64 core topologies [9].</td>
</tr>
<tr>
<td>4.11</td>
<td>Area overhead including router and link area for CTorus and other each topology [9].</td>
</tr>
<tr>
<td>4.12</td>
<td>Speedup of 64 core topologies for (a) SPEC CPU2006 and PARSEC and (b) SPLASH-2 benchmarks [9].</td>
</tr>
<tr>
<td>4.13</td>
<td>Saturation throughput of CTorus and other topologies under synthetic traffic patterns [9].</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

Computers have conventionally used a single core as the central processing unit. The scaling down of silicon technology yields smaller transistor sizes which has led to multiple, smaller and faster processing cores on a single chip. According to Moore’s Law, the rate of silicon technology scaling doubles the number of transistors on a chip every 18 months. The transistors curve in Figure 1.1 shows the approximate doubling of transistors. Since 1975, the number of transistors per chip has been steadily increasing. For example, the Intel’s 8086 microprocessor introduced in 1979 had 20,000 transistors [11]. In 2010, the IBM z196 Quad-core contained 1.4 billion transistors [12]. The frequency curve in Figure 1.1 shows the trend in clock frequency. Clock frequencies increased to allow more pipelining stages which led to higher throughput of single processing cores [13]. The single-thread performance curve in Figure 1.1 shows that the improvement of single core processors directly follows the increase in clock frequency. Single-thread performance improvements continued until only around 2006-2007 when clock frequencies hit 2-3 GHz and both performance and frequencies plateaued. Physical devices did not limit the increase in clock frequency; clock frequencies were purposely limited in order to reduce power consumption [14]. The typical power curve, shown in Figure 1.1, illustrates the increase in processing power per chip. Before 1975, processors consumed less than a watt of power, whereas a 3.2 GHz Pentium 4 Extreme Edition consumed 135 watts of power in 2003 [15]. As power dissipation and heat are becoming major challenges, the clock frequency was reduced to meet power budget constraints. For example, from Figure 1.1, in 2000-2005 clock frequencies as well as power increased by a factor 3. However, from 2005-2010 clock frequencies remained around 2-3 GHz which resulted in no increase in power consumption. In addition to the dynamic power used, static power caused by leakage currents is becoming a concern as transistor sizes become smaller.
Computer architects have improved performance at the microarchitecture level by using instruction level parallelism (ILP) [16, 17]. ILP separates instructions that are not dependent on one another. Independent instructions can be executed in parallel, and thus improve execution time. However, the improvement from ILP has diminishing returns. Since 1986, ILP has steadily increased performance by 52%/year until around 2003 [13]. At that time, the performance was limited by the number of instructions that can be executed in parallel. With all independent instructions, ILP would lead to a linear increase in performance. However, there are typically a limited number of independent instructions in a program code, and the gap between the ideal improvement and actual improvement was at 30:1 in 2003. By 2011, this gap has increased to 1,000:1 and is predicted to increase more.
to 30,000:1 by 2020 [13]. The diminishing returns of ILP, along with thermal problems and the increase in power consumption, limits the performance of single core processors.

Recently, there has been a trend towards integrating hundreds or thousands of smaller cores on a single chip called chip multiprocessors (CMPs) [18, 19]. The bottom curve of Figure 1.1 shows that the increasing number of cores is a trend towards CMPs. Around 2005, the prediction of single core performance limitations along with the availability of transistors has led to the number of cores on a chip to increase. Some examples of current CMPs are Tilera’s 64 core processor [20], the 80-core Intel TeraFlops research chip [21], and a 512-core FERMI chip from Nvidia [22]. Since power is proportional to the frequency and to the square of the voltage, each core in CMPs operates at a reduced frequency and voltage, thereby exploiting parallelism to gain performance improvements.

1.1 NoC

With an increasing number of cores, an efficient communication fabric is required to connect the cores. The design of this fabric will have a significant effect on CMP performance [23–25]. The simplest communication fabric is a bus-based network. A bus is a shared communication line connecting all the cores. This type of communication allows fast point-to-point communication with a small area overhead. However, bus-based networks do not scale well with the growing number of cores. With a large number of cores, contention for the network increases under high network load and the bus becomes a bottleneck. Additionally, with higher clock frequencies, the dissipated power rises and more clock cycles are required for data to traverse the bus.

Network-on-Chips (NoCs) is a novel design paradigm that replaces the traditional bus-based networks to overcome the dual problems of scalability and latency [23–28]. NoCs can provide low power, low latency, and high bandwidth communication for CMPs. Traditional NoCs designs consist of metallic interconnects arranged in topologies such as
2-D meshes or tori [17]. Typically, adjacent cores are connected using short links to reduce latency and power. Cores or groups of cores are connected to the network through ports. Data packets traveling in the network use a series of links and routers to progress to their destination.

### 1.1.1 NoC Router Microarchitecture

The main components of a NoC are the *links* and the *routers*. Links in traditional NoCs are metallic wires with repeaters used to propagate signals between routers. Figure 1.2 shows the router microarchitecture with $n$ inputs and $m$ outputs. The links entering a router are called the inputs, and the ones leaving are called the outputs. The number of inputs/outputs is called the router degree, or radix. Routers route bits of information, or *packets*, to the progressive output port. A packet may contain control information or request/responses for memory management. Routers consist of *buffers*, *crossbars* and control logic for flow control. Buffers store information either at the input, output, or both [15, 17]. Virtual channels (VCs) are multiple buffers at a single input. VCs are used to avoid deadlocking [29] and to improve performance by reducing Head-of-Line (HOL) blocking. With only one buffer, HoL blocking can occur when the first packet is stalled due to some contention. Other packets behind the first packet cannot proceed even if they are not affected by the contention. However, when packets can be allocated to more than one VC, HoL blocking can be avoided.

### 1.1.2 NoC Pipelining and Flow Control

Packets have subdivisions of information called *flits*. Flits can be further divided into *phits* which equal the width of a link, where the width is the number of parallel wire lines constituting the link [17]. A packet contains a header flit that allocates resources (links, buffers, etc.) and contains control information such as destination or data type. The head
Packets are routed through the network in pipelined stages. All of the baseline pipeline stages are shown in Figure 1.3(a). When a head flit first arrives at a router it is stored in the buffer write (BW) stage. This stage writes flits to an input buffer. Next, the correct route is determined in the route computation (RC) stage. The RC block, shown in Figure 1.2, looks into the control information in the header to calculate where the packet needs to go.
Then in the VC allocation (VA) stage, the packet must arbitrate for an input buffer at the downstream router. The VA stage, shown as a block in Figure 1.2, knows the downstream buffer information associated with the packet’s output and allocates the packet to a VC. Once the whole packet wins a buffer slot, each flit then contends for an output port in the switch allocation (SA) stage. The SA, also shown in Figure 1.2, determines which flit can traverse to the inputs and outputs of the crossbar. Once the flit wins arbitration, it moves across the crossbar in the switch traversal (ST) stage. Finally, the flit moves to the next router in the link traversal (LT) stage. Only the head flit must perform the RC and VA stages because subsequent flits follow the head flit until the tail deallocates the VC, as shown in Figure 1.3. However, the body/tail flits must be delayed two cycles for the head to finish the VA and SA.

One of the body/tail flit delays can be eliminated using lookahead routing [30], as shown in Figure 1.3(b). Lookahead routing sends routing information early to the downstream router so that the RC stage can be computed before the flit arrives at the downstream router. However, this still leaves a one cycle delay because the body/tail flits must wait for the head to complete the SA stage. Using speculation techniques [31–33], the VA and SA stages can be performed simultaneously. Speculation assumes that the flit will be able to traverse the switch. This eliminates the one cycle delay, shown in Figure 1.3(c), if this assumption is correct. However, if the network load is high, then the speculation might be incorrect and additional delays will be added.

The flow control determines how resources will be allocated. Some examples of flow control techniques are circuit switching, packet-based, and flit-based flow control [17]. Circuit switching holds resources (i.e. link or buffers) for a certain amount of time even if a packet is not being sent. This type of flow control favors large packet sizes. Packet-based and flit-based flow controls are different because these techniques allocate resources only for one packet or flit, respectively, at a time. The advantage of this is that links and buffers
can be utilized more efficiently. Most NoCs use flit-based flow control over packet-based flow control because routing with a finer granularity increases buffer utilization and reduces blocking time [17]. When a flit moves from router to router, a credit-based system can be used to keep track of buffer information [17]. Credits are sent from the downstream router to the upstream router to indicate when a flit leaves a buffer. For example, the upstream router initially knows how many buffers are available and stores this number. When the upstream router sends a flit, it decrements this number so that it knows how many more flits can be sent before it needs to halt. When a flit leaves the buffer in the downstream router, it sends a credit back to the upstream router and the upstream router increments the number of available buffers.

1.1.3 NoC Topologies

Figure 1.4 shows a 64 core example of three leading NoC topologies where the large squares are cores, small squares are routers, and the lines represent metal wires. The mesh topology shown in Figure 1.4(a) connects cores in a cartesian grid-like fashion. Each core has its own router which routes information and serves as a connection point to the network. The advantage of this type of topology is that the point to point communication to neighboring routers lowers the contention at each router. The disadvantage of the mesh topology is the high network diameter. The maximum number of hops a packet must take from one router to another router in the topology is defined as the network diameter. For example, the network diameter of a 64 core mesh network is 14 hops for the corner-to-corner route. This high network diameter causes high packet latency and degrades performance (throughput and latency) [34]. Figure 1.4(b) shows a concentrated mesh (CMesh) topology in which four cores are concentrated to share a single router [24]. Routers are also connected in a grid-like fashion with the addition of eight longer
Figure 1.3: (a) Baseline pipelining stages, (b) baseline using lookahead routing, and (c) baseline using look-ahead routing and SA speculation

links around the edge of the topology. The concentration of a router reduces the network diameter to four, thereby decreasing packet latency. However, the sharing of the four cores reduces the performance of the network due to contention, hence increasing latency. Figure 1.4 (c) shows the Flattened Butterfly (FBfly) topology which also concentrates four cores [35]. In a FBfly, extra links are added such that the x and y dimensions are fully connected as shown in the figure. These extra links further reduce the network diameter to only two. However, there is a large area overhead due to the additional links which increase the router degree. The router degree is defined as the number of inputs/outputs of a router. As FBfly scales, the router degree increases with x and y dimensions which increases the
Figure 1.4: Leading NoC topologies (a) Mesh, (b) Concentrated Mesh (CMesh), and (c) Flattened Butterfly (FBfly).
area even more. Many commercial applications use a simple mesh topology. The 16-core RAW CMP developed by MIT used a mesh for short latencies between groups of cores [36]. More recently, Intel’s 48-core SCC microprocessor used 24 routers in a 6x4 mesh architecture [37].

1.2 Challenges with Traditional NoCs

The links in traditional NoCs are made of metal wires since this is the most convenient method for current silicon complementary metal-oxide-semiconductor (CMOS) fabrication techniques. However, conventional NoCs require multi-hop communication which causes high latency and power dissipation [38]. Metal wires propagate signals using on-off orthogonal pulses in which a one is represented by a high voltage on the wire and a zero is represented with no voltage. The value of the high voltage is approximately +1 V depending on the CMOS technology node. On-off signaling requires more energy than antipolar which uses positive and negative voltages; however the transmitter and receiver circuitry is simpler. The delay of a wire is given by:

$$\tau = R \times C$$

(1.1)

where $\tau$ is the propagation delay in sec, $R$ is the wire resistance in ohms, and $C$ is the capacitance in farads. As the length of the wire increases, so do the resistance and capacitance. Therefore, the wire delay increases quadratically with length. Repeaters are used in metal wires to propagate signals faster. In addition to the power consumption of the actual metal wire, which increases with the square of the voltage and the wire resistance, there is also power consumption due to the repeaters. The repeaters, which are comprised of transistors, have a power consumption which increases proportionately to the wire capacitance ($C$), switching frequency ($f$), and the square of the voltage ($V$) [15]:

$$P \propto C \times V^2 \times f$$

(1.2)
As CMOS technology scales down, the total power dissipation of metal wires are not scaling proportionally. A 1 mm link in 90 nm technology dissipates 59.4 mW of power compared to 50.6 mW of power in 45 nm technology [39]. Even though the technology node is halved, the power consumption does not decrease by a factor of two. Additionally, with higher clock frequencies more repeaters are needed to propagate signals faster, further increasing power. Voltage and frequency scaling are techniques that can reduce the power dissipation, but they can result in lower performance such as higher packet latency [40]. The combination of the delay and power dissipation of metallic wires limits the performance of traditional NoCs that use many long metal wires. These challenges have demanded the need for emerging technologies such as 3-D interconnects, nanophotonics, radio frequency (RF) transmission lines, and wireless technologies on chip [4–7, 38, 41–48]. The compact nature of 3-D NoCs can offer short and fast communication in the vertical directions. However, the power dissipation of the cores and interconnects concentrated over a small area causes problems with heat [49]. On the other hand, nanophotonic interconnects offer the opportunity to transmit data at high bandwidth with low power consumption. Nevertheless, there are still many challenges with nanophotonic technology such as the layout of optical waveguides and on-chip integrated photonic devices, e.g. fabrication and integration challenges [38]. Another recent alternative to metallic NoCs is wireless NoC. Wireless interconnects can provide highly energy-efficient, complementary metal-oxide-semiconductor (CMOS) compatible, and area-efficient NoCs. Wireless NoCs have been proposed and analyzed in [4, 5] to create low power and low latency inter-router links.

Routers in traditional NoCs are also problematic because of high power dissipation and area overhead. In addition to the links, the buffer and crossbar consume a significant portion of the NoC power dissipation. Researchers have shown that router buffers are responsible for approximately 46% of the total router power [50]. Buffers and crossbars also occupy a substantial NoC area. Crossbars have been shown to occupy approximately
54% of the router area [50]. Since there is a large number of routers at every core or every four cores in most topologies, the router power and area will consume a large portion of the overall chip power and area. Researchers are proposing router optimizing techniques to reduce this power and area overhead. Some buffer optimization research includes moving router buffers to the links by using repeaters to store packets [51] and completely removing router buffers by dropping or deflecting packets [52, 53]. These router optimizations aim to lower the router buffer power and crossbar area. Clearly, both NoC links and routers (buffers and crossbars) will have a significant impact on the power, area, and performance of future CMPs. However, emerging wireless technologies and novel router optimizations are promising solutions for the current problems with NoCs.

1.3 Wireless/RF Technologies

There are three common design methods used in wireless NoC design: (1) hybrid networks, (2) edge-to-edge chip or off-chip communication, and (3) large groups/subnets of cores sharing a wireless hub. A combination of different types of interconnects (metallic, 3D, and nanophotonic) can be used in the same network to create hybrid networks. Hybrid networks often use emerging technologies and metallic interconnects to complement each other in order to improve power, area, or performance. Wireless hybrid networks use wireless links to reduce power, area, and latency and use metallic wires to compensate for the limited wireless bandwidth. The design in [4], shown in Figure 1.5(a), assumes a total bandwidth of approximately 500 GHz is available. Since this is not enough bandwidth for NoCs with a medium or large number of cores, metallic wires are also used. The NoC design in [5], shown in Figure 1.5(b), called WCube is also limited to 500 GHz. Likewise, this design uses wired links to supplement the wireless bandwidth. A scalable architecture is created which uses a wired mesh called a cluster on one tier and a wireless hypercube topology on the second tier.
The second common design method is to use wireless/RF links to communicate all the way across the chip or off-chip. Long wireless/RF links are useful because they can avoid multihop routing which will lower packet latency and power dissipation. Several designs have used an RF transmission line to communicate across chip [6, 54]. The work in [6], shown in Figure 1.5(c), used an RF interconnect (RF-I) laid in a zig-zag pattern on top of a wired mesh. This interconnect propagates signals at the speed of light and is used for edge-to-edge chip communication. On the other hand, the wired links are used for short communication. The work in [55] used an omnidirectional antenna to broadcast the system clock across the chip to all of the cores. The wireless interconnect alleviates the problems of clock skew caused by long global wired links. Additionally, wireless interconnects were used in [56] to create long wireless links between computing chassis. The links used an energy of 2 pJ/bit to transmit a maximum distance of 30 cm. The work in [38] explored the use of short carbon nanotube (CNT) antennas which can be used for edge-to-edge wireless communication. The network in [4], Figure 1.5(a), used CNT antennas in wireless links for long distances across subnets and wires for the short distance communication within a subnet. Additionally, the wireless hypercube network in WCube [5], Figure 1.5(b) transmits long distances to other clusters to avoid multihop wired communication. On the other hand, short range wireless links were used in [7], as shown in Figure 1.5(d), for an irregular NoC topology. However, the short range of the links required additional wireless transceivers to repeat the signal across the chip.

The last common design concept is the use of large groups/subnets of cores sharing a wireless hub. Since wireless interconnects have limited bandwidths and can occupy a large area with current technologies, the number of wireless transceivers must be limited. Therefore, it is common to connect many cores to one centralized wireless hub using metal wires. Whenever a core wants to use a wireless link, it must first take a wired path to the centralized wireless hubs. Two examples of using centralized hubs are in [4] and WCube.
Figure 1.5: Related work: (a) Hybrid Subnets [4], (b) WCube [5], (c) RF-Interconnect [6], and (d) Irregular Wireless [7].

[5]. The cores in each subnet or cluster are one wired hop or more away from the wireless hub. The work in [57] uses four 5x5 subnets each with a centralized hub. Packets are routed to the path with the smallest hop count. These three design concepts will be adapted at the channel level in order to design energy efficient and low latency wireless links.

1.4 Router Optimizations

A large portion of the router power is due to the buffers. Reducing the number of buffers will decrease power and area; however, the performance may also degrade, where
the term performance in this thesis refers to throughput, latency, and speedup of the network. Several designs have been studied to reduce the buffer power overhead while still maintaining performance. The iDEAL design in [51] reduced the number of router buffers in order to reduce the power and sustained performance by using the links as both buffers and links. By essentially moving the buffers from the router to the links, the power normally used for the link traversal can also be used to buffer data while also maintaining enough buffers to sustain performance. The combination of links and buffers are called link buffers, or channel buffers. However, head-of-line (HoL) blocking was caused by the serial link buffers. HoL blocking occurs when packet A is waiting in a buffer due to contention somewhere downstream. Packet B is behind packet A but does not have the same contention. Since the buffers are serially connected, packet B will be blocked until packet A moves. The Elastic Channel Buffers (ECBs) design in [58] eliminates router buffers completely and replaces the link repeaters with flip-flops. ECBs avoid HoL by duplicating the network links. With more than one link between routers, blocked packets can use the other link to move around the head of the line. Another method to remove the buffer power overhead is to completely remove the buffers using a bufferless design. Bufferless networks, such as FlitBLESS [52] and SCARAB [53], use deflection or dropping of packet to resolve contention. However, these networks need to quickly determine which packets move in the preferred direction and which packets are deflected or dropped. Additionally, bufferless networks have high power consumption at high network loads. Crossbar area and power increase quadratically with the number of inputs and outputs. Typically, power and area overhead can be reduced by lowering the number of inputs/outputs or using split crossbars. The RoCo design [59] restricts the direction by row or column. This restriction creates a low radix router to be decoupled into two separate 2x2 row and column crossbars. The authors in [60] used two split crossbars in a high radix router. Half of the input ports are partitioned to one crossbar and the other
half are partitioned to another. Low power channel buffers and small split crossbars will be used at the microarchitecture/topology level to optimize the NoC routers.

1.5 Major Contributions

In this thesis, two approaches are proposed to tackle the problems of NoC architectures at the channel level and microarchitecture/topology level. At the channel level, we leverage wireless technology to reduce link power and alleviate the latency problems of long metal links. At the microarchitecture/topology level, we propose a CTorus topology which moves router buffers to the link to reduce router power and uses split router crossbars to lower the router area overhead.

The proposed architecture in this thesis, called iWISE: inter-router Wireless Scalable Express Channels, uses a combination of the three common design concepts: (1) hybrid networks, (2) edge-to-edge chip or off-chip communication, and (3) large groups/subnets of cores sharing a wireless hub. However, centralized wireless hubs can cause a bottleneck with many cores wanting to use the wireless links. Also, getting to the wireless hubs adds additional hops to the total packet hop count causing additional latency and power dissipation. iWISE is a hybrid network that uses wireless communication links for edge-to-edge chip communication. However, iWISE also uses smaller groupings by distributing wireless hubs and uses wireless links at shorter distances to increase the power savings and reduce hop count.

A CTorus topology is also proposed in a separate network which uses dual channel (dc) buffers and multi crossbars (mx). The design in this thesis uses channel buffers to reduce the significant buffer power consumption. Dual channels are used to improve performance by avoiding HoL blocking. The mx crossbar design in this thesis uses four split crossbars each corresponding to a different routing quadrant. This reduces the area overhead compared to a large unified crossbar. The major contributions of this thesis are:
1. A wired/wireless hybrid network is designed with hierarchical subsections to efficiently share wireless bandwidth and distributed wireless hubs to reduce metal wires in a one-hop 64-core network (iWISE-64) and three-hop maximum 256-core network (iWISE-256).

2. Router optimizations including channel buffers that avoid HoL blocking and split crossbars that reduce power and area are implemented in a concentrated torus (CTorus) topology to improve performance.

3. Wireless links and router optimizations are separately evaluated and improve power, performance, and area under synthetic traffic and real benchmarks such as PARSEC [61, 62], Splash-2 [63], and SPEC2006 [64].

1.6 Organization of Thesis

The work in this thesis is split into two separate but related components. The first component looks into optimizing links by lowering hop count and link power using wireless interconnects. The second component explores reducing the overhead at routers by optimizing buffers and crossbars in a CTorus topology. The organizations of this thesis is as follows. Chapter 2 focuses on wireless interconnects in the proposed iWISE architecture including topology, shared communication using tokens, scalability, and wireless devices. Chapter 3 focuses on buffer and crossbar optimizations in the proposed CTorus topology which uses power efficient channel buffers and low area split crossbars. Chapter 4 presents the methodology used for the performance analysis and shows the results obtained for iWISE and CTorus including power, area, and performance (latency, throughput, speedup, etc.). Chapter 5 concludes the thesis and discusses future work.
2 EMERGING TECHNOLOGY: WIRELESS INTERCONNECTS

This chapter discusses the proposed iWISE design in detail. The first section explains the topology, router microarchitecture, and communication scheme with tokens. A one hop, 64-core architecture called iWISE-64 and a scalable three hop, 256-core architecture called iWISE-256 are shown. The second section discusses the design of wireless transceivers to be used in iWISE.

2.1 Architecture: iWISE-64

Figure 2.1 shows the proposed wireless 64-core architecture called iWISE-64: inter-router Wireless Scalable Express Channel. Solid lines represent metal wired links and dashed lines represent wireless links. Each router is connected to four cores which defines a cluster. Each cluster is connected to neighboring clusters, through wired links, creating a mesh topology. All non-neighboring clusters are connected via wireless links. Using wireless links for non-adjacent clusters utilizes the design concept of using wireless interconnects to transmit long distances. Therefore, the packets can skip over intermediate routers and reach their destination in one hop creating low packet latency and power. Fully connecting this network distributes wireless hubs, avoiding a centralized point, which results a large number of wireless links. However, the router area and wireless bandwidth requirement can be reduced by organizing clusters into groupings and using a token sharing scheme which will be explained next.

To create a scalable and flexible iWISE design, a set of letters \((N, C, S, G)\) is used to define groupings in the architecture. \(N\) stands for the maximum number of cores per cluster, \(C\) is the number of clusters per set, \(S\) is the number of sets per group, and \(G\) is the number of groups in the architecture. Therefore, the total number of cores is the architecture is the product of \(N \times C \times S \times G\). Figure 2.2(a) shows the logical grouping of clusters and sets as well as the addressing convention of iWISE-64. In this thesis, \(N\) will always equal
four since it has been shown that concentrating four cores optimizes performance and
router/buffer area [65]. Also, \( C \) and \( S \) will always be equal to four because this lowers the
token contention while still maintaining flexible communication which will be explained
in the token scheme. \( G \) will equal one in iWISE-64 so that the number of cores equals 64.
The clusters are addressed by the notation \( C(c, s, g) \). Capital letters represent maximum
values and their lower case equivalents represent arbitrary address such that \( c \) is an arbitrary
cluster from 0 to \( C-1 \), \( s \) is a set from 0 to \( S-1 \), and \( g \) is a group from 0 to \( G-1 \). For example,
a cluster addressed \( C(3, 1, 0) \) is cluster number 3 in set 1 in group 0. Note that \( g \) will always
be 0 in iWISE-64 for this thesis; however, it is still shown for clarity and will change in
iWISE-256. Figure 2.2(a) also shows the wired connections between clusters as straight
lines as well as the wireless connections between sets as arrows. Each arrow coming out of
a set is shared between the clusters within the set. This sharing is needed to limit the router
degree and wireless bandwidth.
Figure 2.2: (a) iWISE-64 using the token-partial communication where $C = 4$ and $S = 4$ and (b) communication example from set 0 to set 1 [1].

A token-based sharing scheme is used to limit the wireless bandwidth requirement while providing flexible communication between clusters. At a high level, the role of tokens is to fairly allocate clock cycles in which a cluster can transmit. In other words,
A token gives a cluster the right to transmit on a certain frequency, or wireless link. Since there are four wireless links at each set, there are four tokens that clusters can pass to each other. The physical implementation of a token is four bits of data with only one bit high. The index of this high bit represents a wireless link, hence, the destination set. For example, if C(1,0,0) of set 0 in Figure 2.2(b) has the set 1 token then it can transmit to any cluster in set 1. As it is not possible for any other cluster in set 0 to have this same token, C(1,0,0) will be the only cluster transmitting to set 1. Within each set, the four clusters share tokens which are independent of other set’s tokens. When a packet is being sent to a set, each of the four clusters receives the packet. Then, each cluster will check to see if the destination of the packet matches its own id. Finally, the packet is either accepted or rejected. This technique is called single write multiple read (SWMR). For example, if C(1,0,0) wants to transmit to C(1,1,0) in set 1 then all clusters in set 1, C(0,1,0), C(1,1,0), C(2,1,0), and C(3,1,0), will receive the packet. Only the destination cluster, C(1,1,0), will accept the packet and the others will discard it. This avoids additional hops of a centralized wireless hub by sending the packet directly to its destination. This also alleviates the need to tell the receiving cluster when a packet will be arriving. While SWMR may add some additional power, the trade-off is reduced latency.

A token sharing scheme was chosen because tokens can be quickly passed between clusters so that clusters will not have to wait many cycles for a token. The design choice to have $C=S=4$ was done so that the number of tokens equaled the number of clusters. This gives each cluster an opportunity to transmit during a time frame. When a cluster captures a token, it transmits the data then passes the token along. If no other clusters need this token then it can be passed back to the original cluster and that cluster can transmit again during the next time frame. However, if all the other clusters also need the same token then the original cluster must wait four cycles before it can transmit again. This token sharing scheme is called Token-Partial (TP). Another, less restrictive design is considered
and is used for an additional comparison in the results section. This design ignores set organizations by sharing 16 wireless links between the 16 clusters. Therefore, there are 16 tokens each representing a cluster. This token sharing scheme is called Token-Full (TF). While this approach does not restrict communication, there is a possibility that a cluster can wait up to 16 cycles for a token. However, certain traffic patterns can benefit from this approach since TF allows multiple clusters within a set to communicate to clusters in the same destination set.

A typical router in iWISE-64, shown in Figure 2.3 has four inputs and outputs (I/O’s) for the wired links, four I/O’s for the wireless links, and four I/O’s for the four cores. Four wireless receivers are used at each router so that each of the four sets are able to communicate simultaneously with one cluster. There are four transmitters at each router; although, only one is used at a time. Deadlocking is avoided since iWISE-64 is a one hop network. Although four VCs are still used at each input to avoid blocking and improve performance. Additionally, since iWISE-64 is a one hop network, the router area can be optimized by using two separate, smaller crossbars. A 8x4 input crossbar can be used to direct incoming traffic from the wired and wireless links to the cores. Another 4x8 output crossbar can be used to direct outgoing traffic from the cores to the wired and wireless links. Two separate, smaller crossbars will greatly save area since the crossbar size increases quadratically with the number of I/O’s.

2.2 Architecture: iWISE-256

iWISE can be scaled to 256 cores by replication of iWISE-64 four times. Each iWISE-64 subsection is a group so that $G=4$ in the iWISE-256 architecture as shown in Figure 2.4. In iWISE-256, the $g$ coordinate is used to show to which group a cluster belongs. Higher level organizations and more coordinates can be added to scale iWISE further. In iWISE-256, the wireless hubs become slightly more centralized in order to reduce router
area overhead. This requires slight modifications within each group. First, set routers and group routers are separated to communicate to other sets and groups, respectively. The group routers are distinguished from set routers by labeling them with an asterisk in Figure 2.4. Since in most traffic patterns, there will be high contention for the wireless links, the group routers are evenly spread out to avoid bottlenecks. Second, the wireless link used to communicate to its own set in iWISE-64 has been removed to reduce router area, and replaced by a diagonal wired link so that each router is only one hop away from a group router. For traffic that must leave a group, being one hop away from a group router will lower the latency and power dissipation of this long distance communication path.
Figure 2.4: Scalable architecture for iWISE-256 showing the wireless communication between sets where $C = 4$, $S = 4$, and $G = 4$ [1].

The token sharing scheme in iWISE-256 uses both “set” tokens as well as “group” tokens. There are three set tokens used to transmit to other sets and three group tokens used to transmit to other groups. Therefore, group and set routers cannot communicate directly with wireless links. Inter-group communication will require, at most, one wired hop to a group router plus one wireless hop to another group plus one wired hop to the destination for a maximum three hops. For example, in Figure 2.5 C(0,0,0) is the source and C(1,0,1) is the destination. In step 1a, the packet will take one wired link to the group router, C(2,0,0), in the +y direction. Then in step 2a, it will wait for the group 1 token to
transmit wirelessly to C(2,0,1) in group 1. Finally in step 3a, it will take one more wired hop in the diagonal, or z direction, to the destination, C(1,0,1). Intra-group communication will also require three hops at maximum: one wired hop to a set router, one wireless hop, and one more wired hop to the destination. For example, in Figure 2.5 C(2,1,1) is the source and C(2,2,1) is the destination. In step 1b, the packet will first take a wired hop in the -x direction to C(3,0,1). Once at the set router, it will use the wireless link to transmit to C(0,2,1) when it captures a set token in step 2b. Finally in step 3b, it will take one more wired hop in the +y direction to the destination, C(2,2,1). The shortest path is calculated.

Figure 2.5: Partial view of iWISE-256 with an inter-group and intra-group communication example [1].
from the source and destination header information. Dimension ordered routing is used for wired communication.

The router microarchitecture of iWISE-256 is shown in Figure 2.6. Group routers and set routers both use three wireless I/O’s as opposed to the four in iWISE-64. Group transmitters communicate to other groups and set transmitters communicate to other sets. There is one less wireless link as sets and groups do not communicate with themselves in iWISE-256. An additional wired I/O is used for the diagonal, z-direction, for a maximum total of five wired I/O’s. As before, there are four I/O’s for the four cores in the concentration. The number of wired I/O’s can change and be as low as two depending on its location in the topology. The split crossbar optimization used in iWISE-64 cannot be used again in iWISE-256 because of the three hop nature of the 256 core network. The size of the crossbars in iWISE-256 are 12x12; however, a 256 core FBfly topology would require 18x18 crossbars.

2.3 Wireless Technology

The design of the wireless transceiver must have low power and low area to be an efficient wireless link. Recent wireless transceiver designs have shown energy values of 0.33 pJ/bit [4], 1 pJ/bit [1], 2 pJ/bit [56], and 4.5 pJ/bit [5]. Also, the wireless bandwidth must be large enough to support the wireless link data rates. A bandwidth of 512 GHz is used in [5]. The basic design of a wireless transmitter requires an up-conversion to a radio frequency, a filter, a power amplifier (PA), and an antenna. The wireless receiver requires an antenna, a low-noise amplifier (LNA), a filter, and a frequency down-conversion [66]. Additional components such as a detector can be added to improve the bit error probability or some components can be removed at the expense of bit error probability.

**Modulator/Demodulator:** A modulator converts the digital data to a RF data signal and the demodulator converts the RF signal back to digital data. A common modulation
Figure 2.6: Router microarchitecture of iWISE-256 showing set router and group router [1].

Technique among on-chip transceiver designs is on-off keying (OOK) because the simplicity of the design requires low power and area [67]. Digital data can be modulated onto a RF carrier in a variety of ways including a mixer as in [1, 68] or a Mach-Zehnder optical modulator as in [4]. Silicon Mach-Zehnder modulators can modulate at speeds of 10 Gb/s with low RF power consumption [69]. A mixer will multiply the waveform of the digital data to the waveform of the carrier. The result is a carrier with the digital data modulated onto it which will then be amplified and sent to the antenna.
**Oscillator:** The carrier frequencies are the signals that propagate the modulated data to the receiver and can be created using oscillators. The carrier frequencies can range from around 60 GHz up to the low THz range. Each transmitter requires an oscillator tuned to different frequencies in order to modulate data on different channels. The receiver must need an oscillator tuned to the same frequency as the transmitter in order to correctly demodulate the signal. The design in [70] uses a injection-locked oscillator (IJLO) for its low power of 9 mW and low complexity compared to a conventional phase-locked loop oscillator which has a power of 185.2 mW in [71]. Other types of oscillators such as voltage control oscillator (VCO) can be used based on design specifications such as carrier frequency. A VCO is used in [55] for on-chip wireless clock distribution at 15 GHz.

**PA/LNA:** Power amplifiers (PA) may be needed in the transmitter to amplify the outgoing signal so that the message can be detected at the receiver. The work in [72] uses two cascaded type-A amplifiers that have low power efficiency but operate well in high-frequency wideband applications. Since the distances between the transmitter and receiver may be different depending on location, a tunable PA would be useful to reduce power for shorter distances. Low-noise amplifiers (LNA) may be needed to amplify incoming signals at the receiver when the signal is weak. A three stage, low-power wideband LNA was designed in [68] for a 9.08 mW millimeter-wave transceiver for on-chip wireless applications. A LNA was designed in [73] which created a wireless link for ultra-wideband (UWB) communication. UWB communication uses a range of carrier frequencies that are 20% larger than the center frequency or wider than 500 MHz [74].

**Antenna:** The design of the antenna is a challenging design problem with no simple solution. The small size of chips will limit the antenna to inefficient lengths. Additionally, the small distance between antennas, surrounding metal layers, and the bulk silicon of the chip will significantly reduce radiation efficiency and gain [75]. The work in [4] proposed to use a carbon nanotube (CNT) antenna. However, the integration of CNTs
on chip is challenging. On the other hand, metal wires can be fabricated on the chip to create simple dipole antennas. The work in [76] fabricated two on-chip dipoles in 0.18 μm CMOS technology that occupied an area of 1.5 mm x 1.5 mm. A different approach uses SiGe BiCMOS technology to implement an inter-chip transceiver with bond-wire antennas operating at 43 GHz [72]. However, in most designs the antenna sizes are very large compared to the 20 mm x 20 mm chip. Since frequency is inversely proportional to wavelength, higher carrier frequencies will require small antennas which may fit on chip. The work in [77] uses on-chip transceivers operating in the range of tens to one hundred GHz with zig-zag antennas to reduce area. Different types of on-chip antennas were studied in [75] such as Yagi, rhombic, and loop antennas which operate around 60 GHz.

The transceiver design in [8] is shown in Figure 2.7. The design uses VCO to generate carrier frequencies centered around 100 GHz. Data is modulated onto the carriers using OOK. The mixers in this design use double gate (DG) complementary metal-oxide-semiconductor (CMOS) technology to decrease power consumption and area overhead. The PA is tunable so that the less power can be used to transmit shorter distances. Overall, the energy consumed by the design is 0.9 pJ/bit at a distance of 5 mm and the area overhead for the transmitter and receiver is 0.054 mm² and 0.04 mm², respectively.

For iWISE-64, a total bandwidth of 512 GHz is split into 16 channels each operating at 32 Gbps. Each channel is centered around a different frequency to avoid interference using frequency division multiplexing (FDM). With FDM each transmitter and receiver has a separate set of frequencies to use which creates the 16 separate channels. Each router will be able to transmit on four different channels as there are four different sets in iWISE. However, the token scheme implements time division multiplexing (TDM) so that only one frequency is being used at a time by different routers. TDM separates transmission in the time domain so that frequencies can be re-used without interfering. Therefore, a router can only transmit on one frequency at a time. iWISE-256 may require more bandwidth than
Figure 2.7: Transceiver design implemented in 32 nm RF-CMOS centered around 100 GHz with OOK modulation operating at 5 Gbps for 16 channels [8].

512 GHz. Therefore, techniques such as spatial division multiplexing (SDM) can be used to isolate frequencies between different groups. SDM may require directional antennas or metal separators between groups.

2.4 Challenges with Wireless Technology

In order for wireless links to improve NoC performance, they must: 1) Provide high data rates ($\geq 10$ Gbps), 2) have power- and area-efficient transceivers, and 3) use an efficient multiple access scheme. Providing high data rates is challenging when the frequency spectrum is limited. In order to accompany metal links, wireless links must minimize power consumption and transceiver area. Antennas in the mmwave frequencies are inefficient due to their size in a restricted area. Since some circuits cannot operate at high data rates, serial-parallel conversions will be needed which will add increased power and complexity.
With the limited frequency spectrum, time- and frequency-division must be used; however, with multiple frequencies being close together in the frequency spectrum, the design of the filters is challenging. Spatial division may also be used, but is challenging at mmwave frequencies.

Using wireless links in different frequency bands offers different trade-offs. Three different frequency bands are considered: 50-150 GHz (mmwave), 150-500 GHz, and 500 GHz-3 THz. The low frequency band can offer low power and feasible transceivers, but the data rates for the wireless links are too low and the antennas will be inefficient. Increasing the bands to 150-500 GHz allows low but sufficient data rates and approaches conventional antennas. The design of circuits and devices become more challenging at this band but the possibility of using new technologies such as SiGe-BiCMOS is encouraging. Finally, the 500 GHz-3 THz band allows for ample data rates and reasonable antenna and propagation analysis. However, the circuits and devices are immature at such high frequencies.
3 Router Optimizations

This chapter discusses router optimizations in the buffer and crossbar. First, an explanation of the channel buffer implementation is given. Next, three channel buffer organizations are discussed including dual channel, dual channel multi-input, and single channel multi-input. Two crossbar organizations are shown including dual crossbar and multi crossbar.

3.1 Channel Buffers

Repeaters which are used to propagate signals in conventional metal links can also act as buffers with slight modifications. A conventional link uses inverters every 0.25 mm to quickly propagate a signal down the wire. Repeaters are used to split the wire capacitance into smaller sections so that the capacitance can charge faster resulting in faster signal propagation. A control block and two transistors are added to create a tri-state inverter as shown in the inset of Figure 3.1. The tri-state inverter transmits the signal as a normal repeater would when the control bit is low. However when the control bit is inverted, the output is a high impedance and the input is held on the line. Replacing conventional repeaters with tri-state inverters, also called channel buffers, reduces power and area of the network by decreasing the number of buffers. There is one control block per inter-router link, shown in Figure 3.1(a), which tells the channel buffers when to hold or release data. Using one control block per link lowers power and area compared to [58] which uses one control block per stage. The control block has one output control line for each repeater stage which can store or release data at each stage. The implementation of the control block is shown in Figure 3.1(b). The logic for one control line requires four gates and one flip-flop. The logic gates determine the output for the control line from the inputs of the router and the flip-flop releases the signal on the next clock cycle. Figure 3.1(c) shows the state transition diagram for the control logic. The two states are hold and release. The hold
state sends a signal to the channel buffers to store the flit. When there is congestion at the router and it can no longer accept flits, the congestion signal is sent to the control block and the channel buffers stall the flits until congestion is released. In the release stage, the router sends a signal to the control block and the channel buffer releases the flit. The enable signal is used to reduce the power consumption of the control block. The vc\_en signal indicates when a flit is transmitted to the repeater stage.
3.1.1 Dual Channel Organization

The dual channel (dc) buffer organization of [2] is shown in Figure 3.2. There are two links connecting the upstream router to the downstream router in order to avoid HoL blocking. Additionally, each link has a register at the downstream router: one for input \( I_0 \) and one for input \( I'_0 \). When the input register is full, the control block sends a signal to the tri-state inverters indicating to hold the data. The control buffer uses the DEMUX information to indicate at which channel the flit is arriving via the vc_en signal. When the flit arrives at a register during the BW stage, it proceeds to the RC, VC allocation, SA, ST stage, before moving on to the LT stage. The VC allocation information (0 or 1 for two VCs) is sent with the flit to the DEMUX so that the correct channel is chosen. Once the register is free, the control block indicates the tri-state inverters to release the data and the flits continue on their path. If the channel buffers and registers on both links become full, a signal is sent to the switching control at the upstream router and data transmission is halted. The switching control also tells the DEMUX which input (\( I_0 \) or \( I'_0 \)) to use. The dc buffer organization supports different classes of traffic as well as avoiding HoL blocking and improving throughput.

3.1.2 Dual Channel Multi-Input Organization

The dual channel multi-input (dcM) organization is shown in Figure 3.2(b). This organization has 4 VCs to increase throughput but two channel buffers per VC. There are 4 separate control blocks for the 4 VCs. The switching control determines which VC a flit is allocated according to the congestion information from the 4 separate control blocks. Two sets of 1-to-2 DEMUXes are used to reduce the area overhead compared to one 1-to-4 DEMUX. The VC information, which is 2 bits for the 4 VCs, is sent with the flit to the two
Figure 3.2: Tri-state inverters replacing conventional repeaters in (a) a dual channel (dc) buffer organization, (b) a dual channel multi-input (dcM) organization, and (c) a single channel multi-input (scM) organization [2].
sets of DEMUXes. The goal of dcM is to alleviate congestion for the same input port while minimizing area overhead and saving power.

### 3.1.3 Single Channel Multi-Input Organization

The single channel multi-input (scM) organization is shown in Figure 3.2(c). This design stacks channel buffers near the inputs of the downstream router. This organization reduces congestion, however, HoL can still occur before the channel buffers are stacked. Four control blocks are needed for the four VCs (CB0 to CB3) as well as a fifth control block (CB4) that is fed the congestion information from the other four. CB0 to CB3 send a release signal to CB4 which will allow a flit to be released from the single channel. The flit will be sent to the correct output of the DEMUX based on the VC identifier and when there is no congestion. While this design does not completely avoid HoL blocking because of the single channel, the design reduces area overhead compared to the dcM design. This organization will have a slight performance decrease, however, the area overhead is reduced because of the stacked channel buffers.

### 3.2 Crossbars

The dual inputs of the channel buffer designs are fed to the router crossbar. This section shows two crossbar organizations that utilize the increased throughput from the dual inputs. The two crossbar designs are dual crossbar (dx) and multi-crossbar (mx) organizations.

#### 3.2.1 Dual Crossbar Organization

The baseline crossbar shown in Figure 3.3(a) is a monolithic crossbar. The dual crossbar (dx) organization shown in Figure 3.3(b) splits the baseline crossbar into two separate smaller crossbars. This dual crossbar design differs from other design in [59, 60] in that there is a single register connected to each of the crossbar inputs. This restricts the VC allocation in terms of expected packet direction. For example, with dimension order routing
(DOR), a flit moving in the x direction will always be allocated to the upper VC until it must turn in the y direction. At the turn, the flit will be allocated to the lower VC so that it can use the crossbar which outputs to the y direction. Once in the y direction, the lower VC will always be used. The restricted number of registers limits the VC allocation. However, dx reduces both area and power by using smaller crossbars.

### 3.2.2 Multi Crossbar Organization

The multi crossbar (mx) organization of [2] is shown in Figure 3.3(c) along with a baseline crossbar in Figure 3.3(a). The mx organization reduces area and power by using four smaller, split crossbars. The baseline crossbar is one unified crossbar with 5 inputs/outputs. The mx crossbar takes advantage of dimension ordered routing. With this routing algorithm packets must first travel in the x direction then the y direction. This means that packets traveling in the x direction will either continue in the same x direction or move in the y direction. Traffic in the y direction will continue in the same y direction. This creates four quadrants of traffic: (+x, +y) [North-East], (-x, -y) [South-West], (-x, +y) [North-West] and (+x, -y) [South-East]. Each of the four crossbars is dedicated to one of the quadrants. For example, consider a packet arriving from the I₁ input of the -x direction. This packet can continue in the -x direction and be routed to the O₁ port or move in the -y direction and be routed to the O₃ port. This packet is in the (-x, -y) quadrant. Likewise, if a packet arrives at the I₂ port then it can be routed to O₁ or O₂ port representing the (-x, +y) quadrant. Therefore, there are two output ports for any direction from which the VC allocator can chose. The VC allocator can choose either port when the packet is more than one hop away from its destination. If the packet is exactly one hop away from the destination then it will choose the lower VC so that it can use a crossbar connected to the processing core. Deadlocks are avoided because there are two VCs and the specified
quadrants do not allow circular dependencies. The smaller, split crossbars lower power and area while improving throughput due to multiple output ports.

3.3 CTorus

The proposed CTorus topology which implements the dc and mx organizations is shown in Figure 3.4. The dc and mx designs were chosen because they provide the best overall power, area and performance. In topologies such as mesh and CMesh, traffic is routed towards the center of the network causing congestion. The CTorus topology uses wrap around links, shown as arrows, to balance traffic across the network by allowing packets to move in both +x and -x directions. The wrap around links incur longer delays than short links, but the longer links can avoid additional latency by skipping over the routing stages. Packets are routed in the x direction then the y direction. First, the packet moves in the +x or -x direction depending on the shortest path. Next, the packet can be routed in the +y or -y direction based on the shortest path. The shortest path in either dimension can be calculated based on the x and y coordinates of the source and destination. The network diameter of CTorus is 4 hops since the maximum number of hops in the x direction can be 2 hops plus 2 hops in the y direction. The concentration of cores lowers the hop count and reduces area and power consumption.

Each of the four cores in the concentration logically represents one quadrant ((+x, +y), (+x,-y), (-x,+y), or (-x,-y)) as shown in Figure 3.4. Therefore, each core uses one of the four mx crossbar corresponding to its quadrant. If the traffic from a core needs to move in a direction opposite from its quadrant then it must logically move to another core. However, since the four cores are concentrated there is no one hop penalty. For example, if packet from the (+x,+y) core needs to move in the -x direction then it will use the (-x,+y) crossbar then traverse on the -x link. Likewise, if a packet from the (+x,+y) core needs to move in the -y direction then it will use the (+x,-y) crossbar. Since there are dual links between each
router in every direction, the four cores can improve throughput by using both links. One of the dual links is shared between two cores in a concentration. For example, the (+x,+y) core and (-x,+y) core share the upper +x link and the other two cores share the lower +x link. Also, the (+x,+y) and (+x,-y) cores share the right +y link and the other two share the left +y link. However since the physical implementation uses a concentration, moving to another quadrant does not increase the hop count. Allowing cores to share the dual channels will improve throughput because two packets can be sent in the same direction at the same time. However, packets must still be allocated to the correct VC at turns and when being received by the destination core.

Link and router architecture of CTorus can be implemented with the dc buffer and mx crossbar organizations. The dc organization can be used since there are two links between each router. The four quadrants of the mx crossbar organization can naturally be used with the four quadrant of the cores with an addition of a couple crossbars. Since there are now four cores for one router, the DEMUX and MUX of the mx organization in Figure 3.3 will change to a 4x2 and a 2x4 crossbar as shown in the inset of Figure 3.4. This allows communication to and from all of the cores with a slight area overhead. However, the additional crossbars will only slightly increase the packet power since they will only be used when a packet enters or leaves the network.
Figure 3.3: (a) Baseline crossbar organization, (b) dual crossbar (dx) organization, and (c) multi crossbar (mx) organization [2].
Figure 3.4: CTorus topology using dc buffer organization and mx crossbar organization [9].
4 Results and Discussion

This chapter is separated into two sections. The first section shows performance (speedup and throughput), power, and area for iWISE with wireless interconnects. The second section shows results for the CTorus network with channel buffers and split crossbars.

4.1 Wireless Technologies

In this section, iWISE-256 is compared to other wireless networks including RF-Interconnects (RFI) and WCube as well as electrical wired networks including mesh, concentrated mesh (CMesh), and Flattened Butterfly (FBfly). For an equal comparison, the 100 node RFI network is scaled to 256 nodes by creating a 16x16 mesh and placing the RF hubs at the four corners and center of the topology. iWISE-64 is only compared to electrical networks because the other wireless networks are designed for a large number of cores and would show results similar to electrical networks at 64 cores. In the simulation, each packet consists of four 128-bit flits. For an equal comparison, the bi-sectional bandwidth, the bandwidth between two equal parts of the network, was kept the same for each network. Networks with more bi-sectional bandwidth were given smaller link data rates by adding cycle delays accordingly. Cycle delays were also added in the TP and TF designs to account for the token delay.

For the synthetic traffic patterns, the network load was varied from 10-90% of the network capacity. Measurements were taken after the network was warmed up under low load which injects packets into the network. During the measurement interval, packets were labeled and injected into the network. Once all of the packets reached their destination, the simulation ended. Two different types of synthetic traffic were used: Random traffic, in which packets are randomly assigned sources and destinations, and Permutation patterns where packets had a fixed destination based on different permutations. The networks were
Table 4.1: Core and cache parameters used while extracting traces from SIMICS with GEMS [1].

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>L1/L2 coherence</td>
<td>MOESI</td>
</tr>
<tr>
<td>L2 cache size/assoc</td>
<td>4MB/16-way</td>
</tr>
<tr>
<td>L2 cache line size</td>
<td>64</td>
</tr>
<tr>
<td>L2 access latency (cycles)</td>
<td>4</td>
</tr>
<tr>
<td>L1 cache/assoc</td>
<td>64KB/4-way</td>
</tr>
<tr>
<td>L1 cache line size</td>
<td>64</td>
</tr>
<tr>
<td>L1 access latency (cycles)</td>
<td>2</td>
</tr>
<tr>
<td>Core Frequency (GHz)</td>
<td>5</td>
</tr>
<tr>
<td>Threads (core)</td>
<td>2</td>
</tr>
<tr>
<td>Issue policy</td>
<td>In-order</td>
</tr>
<tr>
<td>Memory Size (GB)</td>
<td>4</td>
</tr>
<tr>
<td>Memory Controllers</td>
<td>16</td>
</tr>
<tr>
<td>Memory Latency (cycle)</td>
<td>160</td>
</tr>
<tr>
<td>Directory latency (cycle)</td>
<td>80</td>
</tr>
</tbody>
</table>

evaluated on the following random traffic: uniform random (UN) and non-uniform random (NUR). UR traffic randomly directs traffic to destinations with equal probability and NUR traffic directs 25% of the traffic to one destination to create a hot spot while the rest of the traffic is uniform random. The following permutation patterns were also used: bit-reversal (BR), butterfly (BFLY), complement (COMP), matrix transpose (MT), perfect shuffle (PS), neighbor (NBR), and tornado (TN).
For the real application traffic, the full execution-driven simulator SIMICS from Wind River [78] was used with the memory package GEMS [79]. The real application benchmarks were executed on SIMICS and the traffic traces were extracted. These traffic traces were then run on the different electrical and wireless networks using a cycle accurate simulator. The following benchmarks were used: SPLASH-2 [63], PARSEC [61, 62], SPEC CPU2006 [64]. The 64 core networks were run under synthetic and real traffic and the 256 core networks were run under synthetic traffic. Table 4.1 shows the parameters assumed for the SIMICS full system simulator. A cycle delay of 2 cycles, 4 cycles, and 160 cycles were assumed to access the L1 cache, L2 cache, and main memory, respectively. For Splash-2 traffic, the following applications were used: FMM (16K particles), FFT (16K particles), LU (512×512 with a block size of 16×16), Radiosity (LargeRoom), Radix (1 Million integers), Raytrace (Teapot), Ocean (258×258), and Water (512 Molecules). Seven PARSEC applications with medium inputs (blackscholes, facesim, ferret, fluidanimate, freqmin, streamcluster, and swaptions) were used and three workloads from SPEC CPU2006 (bzip, gcc base, and hmmer). Orion 2.0 [39] and Synopsys Design Compiler were used to estimate power and area for electrical components. The wireless transceiver energy was estimated to be 1 pJ/bit and area was estimated to be 0.0854 mm² from [4].

4.1.1 Performance

Figure 4.1 shows the speedup for the each of the real application benchmarks. The speedup measures how many times faster an application finished executing on a given network compared to the mesh network. In other words, speedup equals the reference network execution time divided by the test network execution time. Averaged over all the applications, iWISE-TP executes the applications 2.3 times faster than mesh. The high average hop count of the mesh network compared to the one hop iWISE network increases the mesh packet latency because of intermediate routers. The highest speed-up of 2.8
over mesh was for the LU application of the SPLASH-2 benchmark. Applications such as LU allow packets to reach their destination in one hop using the fast wireless links. The lowest speed-up was for the gcc base application of the SPEC CPU2006 benchmark and was measured to be 1.9 over mesh due to token contention in this application. iWISE-TP performs an average of 45% better than FBfly for all benchmarks. The two hop FBfly has fewer intermediate routers than mesh; however, there are still more hops than iWISE which causes iWISE-TP to perform better. iWISE-TP has an average speed-up 42% higher than iWISE-TF. Since there are 16 clusters sharing the same token in iWISE-TF, it is possible that a token can wait up to 16 cycles to receive a token. iWISE-TP uses sets to restrict communication, however only four clusters share a token which means a cluster will only have to wait four cycles at most.

Figure 4.2 shows the saturation throughput of the 64 core networks for different synthetic traffic patterns. Throughput is a measure of how many flits are received per core per cycle. Saturation throughput is the maximum throughput when the offered traffic
injection rate is high. The results shown in Figure 4.2 are relative to the mesh network. While the average hop count of mesh is high causing a poor speed-up, the throughput can be comparable or even better than concentrated networks. In a mesh, each core has its own router; whereas, concentrated networks have four cores contending for a router. The CMesh network has lower performance than mesh in traffic patterns such as UN and NUR. In these traffic patterns, there is high contention at the routers because each core sends to every destination. CMesh has a higher throughput in traffic patterns such as BR, BFLY, COMP, MT, and PS. The lower diameter of CMesh compared to mesh causes higher throughput due to low average hop count and low contention. FBfly has a higher throughput than mesh and CMesh for most traffic patterns including UN, NUR, COMP, MT, and PS. The FBfly network does have concentration but has an even lower diameter than CMesh and mesh. The long links in FBfly allow more traffic to be received for most traffic patterns. On average, iWISE-TP has a throughput 2.4× higher than mesh and outperforms all other networks for most traffic patterns. BR and COMP traffic show the highest throughput of almost 4.5× higher than mesh. UR and NBR traffic show the lowest throughput which is comparable to mesh. iWISE-TF and TP are one hop networks which allows much higher throughput because packets spend less time in buffers. Additionally, the packets spend less time on the network because the wireless links do not have delays attributable to long distance like metal links. In traffic patterns where the throughput is comparable to that in the wired networks, more metal links than wireless links are used. iWISE-TP performs approximately 29% better than iWISE-TF on average for all traffic patterns. The differences in TP and TF are due to the differences in token delay and token contention. If four clusters want to communicate to mutually exclusive sets then there is no contention for a token in TP and TF. However, the token in TF must travel to all 16 clusters before returning to the original sender causing TP to outperform TF. On the other hand, if four clusters want to communicate to the same set then the four clusters must all contend for the same token in
TP; whereas, TF will have little to no token contention but still have a long token delay causing TP and TF to perform similarly as in COMP traffic.

Figure 4.3 shows the average network latency and throughput for iWISE and other electrical networks. As shown, the performance gains of our design depends on the traffic pattern of the network. For both matrix transpose and complement traffic patterns, iWISE shows significant improvement over all other networks. Additionally, the differences in performance between TP and TF is due to the difference in token delay and their restrictive communication. While TF has unrestricted communication in terms of which set it can transmit to, it is restricted by the delay for the token to be passed through all 16 routers. As shown in Figure 4.3(c) and Figure 4.3(f), TP performs significantly better than TF. This is because with matrix transpose traffic, each cluster within the same set is sending to different sets. Therefore, there is no contention in TP or TF and the difference in performance is due to the smaller token delay in TP. In complement traffic, each cluster within the same set wants to send data to different clusters in the same set. This causes a high contention in TF which is why the network saturates at a lower point compared to matrix transpose. There is still no contention in TP, although there is still a significant token delay. The latency plots show TF with a higher latency before saturation compared to TP because the token delay for TF is generally 4X higher. Under uniform traffic, iWISE does not perform as well as FB. This is due to the random nature of uniform traffic. There is contention in both TF and TP. A cluster may have to wait a long time to receive a token because every cluster will need to communicate to multiple clusters in multiple sets. This performance is the trade-off for the power and area reductions.

Figure 4.4 shows the saturation throughput for 256 core networks under synthetic traffic patterns. iWISE-256 shows similar saturation throughput results as iWISE-64. On average, iWISE-256 has a relative throughput of 2.32 for all traffic patterns. The highest throughput is in the BR and MT traffic patterns. The lowest throughput is for UN traffic.
Figure 4.2: 64 core saturation throughput for synthetic traffic patterns: Uniform (UN), Non-Uniform Random (NUR), Bit Reversal (BR), Butterfly (BFLY), Complement (COMP), Matrix Transpose (MT), Perfect Shuffle (PS), Neighbor (NBR), and Tornado (TN).

The improvement of iWISE-256 over mesh is due to the long wireless links and three hop diameter network. iWISE-256 outperforms the second best network, WCube, by 33% averaged over all traffic patterns. iWISE-256 is able to have better performance because of the distributed wireless hubs in iWISE as opposed to the centralized hubs in WCube and RFI. iWISE-256 does not have a wireless link for communication within a set as in iWISE-64. This causes more wired hops especially in traffic patterns such as UR which results a slight decrease in improvement.
Figure 4.3: (a)-(c) Average network latency per packet and (d)-(f) throughput
Figure 4.4: 256 core saturation throughput for synthetic traffic patterns: Uniform (UN), Non-Uniform Random (NUR), Bit Reversal (BR), Butterfly (BFLY), Complement (COMP), Matrix Transpose (MT), Perfect Shuffle (PS), Neighbor (NBR), and Tornado (TN) [1].

4.1.2 Power

The power dissipation for electrical components was estimated from Orion 2.0 for this section. A 128 bit metal link in 32 nm technology for a 1 GHz clock was estimated to dissipate 23.04 mW/mm. The wireless links operate at 32 Gbps and transmit a flit in four clock cycles. Therefore, the wireless links in all networks dissipate a power of 128 mW for a 128 bit flit based on energy of 1 pJ/bit. The buffer and crossbar power dissipation
was 1.39 mW and 13.84 mW, respectively. Each network was tested in a cycle accurate simulator and the total number of wired and wireless link, buffer, and crossbar traversals were counted and multiplied by their corresponding power dissipation.

Figure 4.5 shows the power dissipation per packet for the 64 core networks. iWISE-TP and TF save approximately 55% on average over all other networks. The power dissipation per packet for each network is directly related to the number of hops a packet must go through from source to destination. The highest diameter network, mesh, has the highest average power dissipation followed by the next highest, CMesh, then FBfly and finally the one hop iWISE-64 networks. For uniform traffic where 80% of the link traversals are wireless, iWISE-TP and TF save approximately 63% power over mesh. For complement where 100% of the link traversals are wireless, iWISE-TP and TF save 73% over mesh. In addition to the power savings from being a one-hop network, there is also a savings due to the high number of low power wireless links. In traffic patterns such as BFLY where there is high neighbor to neighbor traffic, iWISE-TP saves 45% over mesh. The low wireless link utilization in BR causes slightly less savings in power. There is little difference in iWISE-TP and iWISE-TF because the packets will take the same paths and the power dissipation will slightly differ because of the token power dissipation.

The power dissipation per packet for the 256 core networks is shown in Figure 4.6. Overall, iWISE-256 saves 43% over wired networks. The highest power savings for iWISE-256 is in COMP traffic and the lowest power savings is in BFLY traffic. The pattern is very similar to the 64 core networks because of the diameters of the networks. Additionally, there is a 20% power savings over the wireless networks. This power savings can be attributed to the design of iWISE-256 which uses more distributed wireless hubs than RFI and WCube. The wireless hubs are distributed so that a packet consumes less power using a wireless link compared to an equivalent wired path. For the transceiver energy used in this thesis, a wireless link is taken for packets traveling more than 2 hops.
Figure 4.5: Average power per packet of 64 core topologies under synthetic traffic loads.

iWISE-256 uses wireless links at medium and long distances which saves power compared to other wireless networks which only use wireless links at long distances.

4.1.3 Area

The wired link area was estimated from Orion 2.0 to be 0.0064 mm$^2$ per mm for a 32-bit link in 32 nm technology. For a wireless link including modulators, demodulators, LNAs, and antenna, the area of 0.0854 mm$^2$ was used [4]. A 5x5 router was estimated to be 0.0519 mm$^2$. Figure 4.7(a) shows the total network area estimations for the 64 core networks. iWISE-64 saves approximately 42% over mesh, 39% over FBfly, and 12% over
CMesh. iWISE-TP and iWISE-TF occupy the same area because the only difference is how the tokens are passed. The total area of the networks is broken down into router and wired/wireless link area. With the wireless transceiver area assumed, the total wireless link area is a small fraction, about 3.5%, of the total iWISE-64 area. This decrease in link area of iWISE-64 contributes to the total area savings. Additionally, the router area of iWISE-64 is less than other networks. The router area overhead of iWISE-64 is 8% less than the router overhead of CMesh and 40% less than mesh. In contrast to the large, unified crossbar required in the other networks, the one-hop iWISE-64 can use two smaller, split crossbars.
Figure 4.7: Area overhead of routers, wired, and wireless links for (a) 64 core networks and (b) 256 core networks.

to decrease router area overhead. The 8x4 and 4x8 routers of iWISE-64 occupy an area of 0.143 mm². This is a 50% savings compared to a unified 12x12 crossbar that would otherwise be needed. Figure 4.7(b) shows the area overhead of each 256 core wired and wireless network which is also broken down into router and wired/wireless links. iWISE-256 is 22% larger than WCube and approximately 8% smaller than RFI. The link area overhead of iWISE-256 is comparable to or less than that of the other networks. However, the router area of iWISE-256 is much higher than other networks. This is because a large, unified crossbar is needed since iWISE-256 is not a one hop network. Additionally, the distributed wireless hubs increases the router input/output ports. The increase in area is the trade-off for the performance and power improvements shown in the previous two sections.

4.2 Router Optimizations

In this section, three channel buffer organizations are evaluated: dual channel (dc), dual channel multi-input (dcM), and single channel multi-input (scM). The two crossbar
organizations are also evaluated: dual crossbar (dx) and multi crossbar (mx) and. These router optimizations are compared to a baseline router with 4 VCs per input and each VC has 4 flit buffers. Each router uses a 4-stage router pipeline. A packet size of four 128-bit flits was used. The dc and mx organizations are evaluated in detail and implemented in a CTorus topology which is compared to mesh, CMesh, and FBfly. Power, area, and timing values are estimated using the Synopsys Design Compiler using both the 40 nm and 65 nm TSMC technology libraries with a voltage of 1.0 V and a frequency of 1 GHz. Bi-sectional bandwidth was kept equal for all designs.

4.2.1 Power

The power for each segment of a link with a repeater is given by: $$P_{segment} = P_{dynamic} + P_{leakage} + P_{short-ckt}$$ where $P_{dynamic}$ is the switching power, $P_{leakage}$ is the power due to the subthreshold leakage current and $P_{short-ckt}$ is the power due to the short-circuit current. The total power dissipation for a flit traversal is calculated by multiplying the power per segment by the number of segments and link width. The tri-state buffer adds additional power due to the capacitance from the extra transistors. Additional power is consumed by the control blocks controlling the channel buffers. The inter-router distance was assumed to be 5 mm for the concentrated networks. To save area, 4.5 mm of the link was assumed to be a single channel then the link was split into dual channels for the remaining 0.5 mm.

Table 4.2 shows the power breakdown for each channel buffer and crossbar combination in 65 nm technology. The power is split into buffer and crossbar power with the link power per mm included in the buffer power. The majority of the power savings of the organizations over the baseline is due to the buffer power. The buffer power is similar for each design with the differences due to muxes/demuxes. The crossbars also save power over the baseline with the mx crossbar having the lowest power. Overall, the dc-mx organization has the lowest power dissipation.
Table 4.2: Power estimation per mm link using Synopsys Design Compiler with the 65 nm TSMC technology libraries at 1.0 V and 1 Ghz clock [2].

<table>
<thead>
<tr>
<th>Design</th>
<th>Power (mW)</th>
<th>%</th>
<th>Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Buffer (with link) + xbar</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>61.32 + 13.56</td>
<td></td>
<td>-</td>
</tr>
<tr>
<td>dc-dx</td>
<td>39.63 + 8.19</td>
<td>-35</td>
<td>-</td>
</tr>
<tr>
<td>dc-mx</td>
<td>39.63 + 5.95</td>
<td>-39</td>
<td>-</td>
</tr>
<tr>
<td>dcM-dx</td>
<td>39.71 + 8.19</td>
<td>-35</td>
<td>-</td>
</tr>
<tr>
<td>dcM-mx</td>
<td>39.71 + 5.95</td>
<td>-39</td>
<td>-</td>
</tr>
<tr>
<td>scM-dx</td>
<td>39.81 + 8.19</td>
<td>-34</td>
<td>-</td>
</tr>
<tr>
<td>scM-mx</td>
<td>39.81 + 5.95</td>
<td>-38</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 4.8 shows the power dissipation for a flit to traverse the baseline and dc-mx organizations for both 65 nm and 40 nm technologies. The power is broken down into link, internal wiring, register, crossbar, control block, demux and mux power. The majority of the power dissipation is due to the inter-router links. At 65 nm the link power is 180.0 mW for both designs, and this reduces by 19% at 40 nm. The link power is the same for both the baseline and dc-mx because the inter-router distance is equal in both designs. The internal wiring, which is the second largest portion, accounts for the wiring between inputs/outputs and the crossbar. Overall, the baseline consumes a total of 232.2 mW at 40 nm due to link power (145.4 mW), internal wiring (63.1 mW), registers (15.5 mW), crossbar (8.0 mW), and demux/mux (0.18 mW). The dc-mx organization reduces the total power consumption by 29% at 40 nm consisting of link power (145.4 mW), internal wiring (13.9 mW), registers (1.7 mW), mx crossbar (3.7 mW), control block (0.05 mW), and demux/mux (0.08 mW).
The leakage power of the baseline was 1.659 $\mu$W and the dc-mx leakage power was 1.525 $\mu$W.

The power dissipation per packet of the CTorus topology with the dc-mx organization is shown in Figure 4.9. CTorus is compared to mesh, CMesh, and FBfly on different synthetic traffic patterns including complement, butterfly, and uniform random. For each traffic pattern, CTorus consumes the least amount of power. For complement traffic, shown in Figure 4.9(a), CTorus saves 25.3% power over FBfly. The link power is approximately similar for each topology; however, CTorus has a significantly lower crossbar and buffer power consumption. This is also the case for butterfly traffic shown in Figure 4.9(b) where CTorus saves at least 42%, and uniform random traffic in Figure 4.9(c) where CTorus saves at least 27.7% over all other topologies. As the size of a crossbar increases, the number of wire taps and the distance a packet must travel from input to output increases. Therefore, the smaller mx crossbar saves power compared to one large crossbar. For example, a large 8×8 crossbar used in CMesh consumes 56.89 mW of power. The small 3×3 crossbar used by a packet in the mx crossbar of CTorus only consumes 14.05 mW. Moreover, the channel buffer power used in the dc organization is 0.205 mW which is considerably less than the baseline register power of 1.74 mW. Therefore, at each intermediate router CTorus saves significant power due to the mx crossbar and dc buffers.

The energy-delay product (EDP) mesh, CMesh, and FBfly relative to CTorus is shown in Figure 4.10 for different synthetic traffic patterns. The EDP can analyze how energy efficient a network is at lowering packet latency. CTorus has an average EDP 17.9% less than CMesh, 15.4% less than mesh, and 21.1% less than FBfly for all traffic patterns. The large diameter network of mesh increases packet latency at a high energy cost. Even though the 5×5 router crossbars are somewhat small, the large number of intermediate routers seen in most traffic patterns increases the energy cost. CMesh can balance latency and energy
more efficiently resulting in a slightly lower EDP than mesh and FBfly. CMesh has a lower network diameter than mesh, lowering packet latency, and a smaller router crossbar than FBfly which lowers the energy cost. FBfly has the largest router crossbar but the lowest network diameter. Finally, CTorus has the same diameter as CMesh; however, the low energy crossbars improve the EDP of CTorus for most traffic patterns with an average improvement of 18% over CMesh.
Figure 4.9: Power per packet for different traffic loads. (a) Complement (b) Butterfly, and (c) Uniform Random [9].
Figure 4.10: Relative Energy-Delay Product (EDP) of CTorus and other 64 core topologies [9].

4.2.2 Area

Table 4.3 shows area breakdown of each organization for 65 nm technology. The dc-mx organization had the lowest area overhead and was estimated to be 0.297 $mm^2$. The baseline area including 2 VC buffers and a $5\times5$ crossbar was found to be 0.284 $mm^2$. The slight increase in area of dc-mx is due to the increase in link area of the dual channels. Although the dc-mx organization occupied more area than the baseline, when implemented in a CTorus topology there is a significant savings in overhead. The router and link area of each topology is shown in Figure 4.11. CTorus saves 68% over FBfly, 53% over CMesh,
Table 4.3: Area estimation for buffers with 1 mm links and crossbars using Synopsys Design Compiler with the 65 nm TSMC technology libraries [2].

<table>
<thead>
<tr>
<th>Design</th>
<th>Area ($mm^2$)</th>
<th>% Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Buffer (with link) + xbar</td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>0.248 + 0.0356</td>
<td>-</td>
</tr>
<tr>
<td>dc-dx</td>
<td>0.272 + 0.0246</td>
<td>+4</td>
</tr>
<tr>
<td>dc-mx</td>
<td>0.272 + 0.0237</td>
<td>+4</td>
</tr>
<tr>
<td>dcM-dx</td>
<td>0.274 + 0.0246</td>
<td>+5</td>
</tr>
<tr>
<td>dcM-mx</td>
<td>0.274 + 0.0237</td>
<td>+5</td>
</tr>
<tr>
<td>scM-dx</td>
<td>0.274 + 0.0246</td>
<td>+5</td>
</tr>
<tr>
<td>scM-mx</td>
<td>0.274 + 0.0237</td>
<td>+5</td>
</tr>
</tbody>
</table>

and 62% over mesh. The link area of CTorus is slightly more than CMesh and FBfly because of the dual channels. However, there is a large reduction in area due to the router. CMesh and FBfly have an increase in router size compared to the baseline due to the high radix routers used in these topologies. While mesh implements the baseline 5×5 crossbar, it also does not use concentration which increases the total number of routers needed by a factor of four. Additionally, area is saved because CTorus reduces the number of VCs by using small channel buffers. Although the dc-mx design occupies slightly more area than the baseline 5×5 crossbar, the ability to be used in a concentrated network lowers overall area overhead.

4.2.3 Timing

The latency for the mx design was 0.24 ns in 40 nm technology which is under our clock period of 0.5 ns. The critical path of the mx design was due to the logic in the VA
Figure 4.11: Area overhead including router and link area for CTorus and other each topology [9].

stage. The dc design had a latency of 0.31 ns due to the channel buffer latency of 0.17 ns, the register buffering of 0.07 ns, and the demux latency of 0.07 ns. The critical path the baseline design was 0.22 ns, and was also due to the VA stage.

4.2.4 CTorus Speed-up and Throughput

Figure 4.12 shows the speedup of CTorus and other topologies for the SPEC CPU2006, PARSEC, and SPLASH-2 benchmarks. For SPEC CPU2006, CTorus has a speedup over mesh of 1.91 for the hmmer application and a speedup of 1.70 for the gcc base application.
with an average of 1.79 for all applications. For PARSEC, the speedup of CTorus ranges from 1.62 for ferret to 1.97 for fluidanimate with an average of 1.79 for all applications. The communication patterns of the SPLASH-2 applications give CTorus an average speedup of 1.9 ranging from 1.70 for FMM and 2.11 for barnes applications. CTorus improves over mesh because the 4 hop diameter of CTorus is less than that of mesh which is at 16 hops. The long links of CTorus reduce intermediate routers which lowers packet latency. The lowest diameter network, FBfly, has the best speedup for all benchmarks because packet latency can be decreased the most. CMesh has the same diameter as CTorus which results in similar performance in light load applications such as blackscholes, fluidanimate, and freqmine. However for heavy traffic loads such as facesim, ferret, and streamcluster, the dual links of CTorus results in better performance by reducing channel contention.

Figure 4.13 shows the saturation throughput on the synthetic traffic patterns for concentrated Torus (CTorus) and other topologies. CTorus has a saturation throughput $2\times$ higher than CMesh and FBfly for the Complement traffic pattern. The dual links in the torus and long wrap around links reduce contention for packets traveling all the way across the chip as in Complement traffic. In traffic such as Butterfly where traffic travels halfway across the chip, the 10 mm links in CMesh and FBfly allow for a 1.4$x$ and 1.2$x$ increase in throughput compared to CTorus, respectively. In Bit Reversal the bits in the address of the core are switched and many cores will communicate with themselves because of symmetric bit addresses. This type of communication does not use the network as much which results in CMesh and CTorus having a similar saturation throughput. However, the communication that does use the network has less contention in CTorus because of the dual links between routers. These links in CTorus improve the saturation throughput slightly over CMesh, approximately 10% over FBfly, and 44% over mesh. Other traffic patterns with a mix of short, medium, and long range communication have comparable results between CTorus,
CMesh, and FBfly with torus having a slight advantage in certain traffic patterns such as Non-uniform Random, and Tornado.
Figure 4.12: Speedup of 64 core topologies for (a) SPEC CPU2006 and PARSEC and (b) SPLASH-2 benchmarks [9].
Figure 4.13: Saturation throughput of CTorus and other topologies under synthetic traffic patterns [9].
5 Conclusion and Future Work

The trend towards integrating hundreds or thousands of cores on a single chip will require an efficient network to communicate among cores. However, traditional NoC topologies, such as meshes or tori, are still limited by high latency and excess power dissipation caused by the metallic NoC interconnects and intermediate routers. The work in this thesis overcomes the limitations of conventional NoCs by focusing on two main components: (i) wireless interconnects and (ii) router (buffer and crossbar) optimizations. Wireless interconnects are used in the iWISE topology. iWISE uses power efficient OOK wireless transceivers to communicate across chip in a one hop 64 core architecture and a scalable 256 core architecture. FDM and a token sharing scheme are used to share communication among the cores. iWISE-64 and iWISE-256 are compared against leading electrical and wireless topologies. iWISE shows an average $2.3\times$ speedup on real application loads, a 55% savings in power, and up to a 42% area reduction. Router optimizations are studied on a separate CTorus topology using a power efficient dual channel (dc) buffer design and a small, multi-crossbar (mx) organization. The CTorus topology uses dual channels and long wrap around links to quickly move traffic. The CTorus design improves speedup by up to $1.9\times$ on real benchmarks, reduces area by 58%, and improves EDP by 21.1%.

The work in this thesis focuses on core to core communication. However, this thesis does not consider using wireless interconnects for core to memory. Since memory communication can become a bottleneck with many cores, wireless interconnects can be used to create a high bandwidth, low power, and low latency link to memory. Another application of wireless interconnects is for collective communication for cache coherence applications. Omnidirectional antennas are suited for the broadcast and multicast forms of collective communication. Finally, reconfigurable wireless interconnects can be used.
to further improve performance by adjusting link widths and topologies to suit certain applications.
REFERENCES


Thesis and Dissertation Services