Design Tradeoffs for Tiled CMP On-Chip Networks

James Balfour*, William J. Dally
Computer Systems Laboratory
Stanford University
{jbalfour,billd}@cva.stanford.edu

ABSTRACT

We develop detailed area and energy models for on-chip interconnection networks and describe tradeoffs in the design of efficient networks for tiled chip multiprocessors. Using these detailed models we investigate how aspects of the network architecture including topology, channel width, routing strategy, and buffer size affect performance and impact area and energy efficiency. We simulate the performance of a variety of on-chip networks designed for tiled chip multiprocessors implemented in an advanced VLSI process and compare area and energy efficiencies estimated from our models. We demonstrate that the introduction of a second parallel network can increase performance while improving efficiency, and evaluate different strategies for distributing traffic over the subnetworks. Drawing on insights from our analysis, we present a concentrated mesh topology with replicated subnetworks and express channels which provides a 24% improvement in area efficiency and a 48% improvement in energy efficiency over other networks evaluated in this study.

1. INTRODUCTION

Chip multiprocessors (CMPs) use increasing transistor budgets to integrate multiple processors on a single die [17]. Tiled architectures provide a scalable solution for managing design complexity and effectively using the resources available in advanced VLSI technologies [20]. The complexity of designing efficient, scalable on-chip communication mechanisms will continue to increase as advances allow more processors to be integrated on a single die; furthermore, power dissipation and wire delay will continue to increase in significance as limiting design constraints [8, 9]. On-chip networks address many of the difficulties associated with managing the increasing complexity of on-chip communication: their modular construction facilitates design reuse and allows high-performance circuits to be used in the interconnect; the structured wiring allows the abundant wire resources to be efficiently used while maintaining well controlled electrical characteristics [4].

This work investigates the design of area-efficient and energy-efficient on-chip networks for tiled CMP architectures. Previous work has described efficient router architectures for on-chip networks [21, 15, 11]. We address the broader issue of the complete network's performance and efficiency. We develop comprehensive analytical area and energy models for on-chip networks. The energy models extend those presented in [21, 7] by incorporating the impacts of physical design and layout on energy dissipation. Our area and energy models incorporate design space attributes including topology, routing, channel design, switch architecture, and buffer sizing with sufficient detail to explore a broad design space. We demonstrate that using multiple on-chip networks to increase bandwidth and path diversity substantially improves performance without adversely affecting area or energy efficiency as measured by the area-delay product and energy-delay product, respectively. Drawing on insights from our analysis, we propose a *concentrated mesh* architecture which offers a 24% improvement in area efficiency and a 48% improvement in energy efficiency over other architectures evaluated in this work.

The remainder of this paper is organized as follows. In Section 2 we present network performance and technology models. In Section 3 we detail the motivating system architecture and constraints imposed by the VLSI technology. In Section 4 we present a canonical network architecture and develop our energy and area models. In Section 5 we describe the design space and present our architectures. In Sections 6 and 7 we describe evaluation methodologies and present results and analysis. We conclude in Section 8.

2. PRELIMINARIES

We begin by presenting a simple analytical model that provides insight into interconnection network performance and will motivate later design decisions. A complete treatment of the subject may be found in [5]. Because management of wire resources is critical to the design of efficient on-chip networks, we then proceed to describe wire models used in our analysis. We conclude this section by describing the technology models used in our evaluation.

2.1 Network Performance

Capacity and latency are two measures which are commonly used to compare networks. Capacity, defined as the injection bandwidth offered to each of the N terminal nodes for uniform random traffic, can be calculated from the bi-

*Supported by the Cadence Design Systems Stanford Graduate Fellowship

ICS06, June 28-30, Cairns, Queensland, Australia.

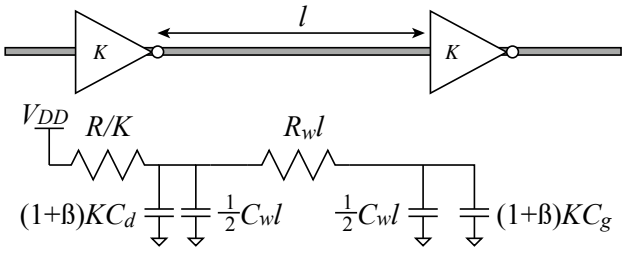


Figure 1: First-Order Repeated Wire Delay Model

section bandwidth (B_{B}) or the bisection channel count (B_{C}) and the channel bandwidth (b) as:

$$\text{Capacity} = \frac{2B_{\mathrm{B}}}{N} = \frac{2bB_{\mathrm{C}}}{N} \tag{1}$$

As the channel bisection provides an indicator of global wiring requirements, equation (1) can be interpreted as expressing a tradeoff between network performance and implementation complexity.

The average contention-free latency (T_0) from source s to destination d depends on a number of parameters, including: the average hop count (H) from s to d and router traversal latency (t_{r}), which determine the routing latency; the channel traversal latency (T_{c}); and the packet length (L) and channel bandwidth (b), which determine the serialization latency (T_{s}). We calculate T_0 as:

$$T_0(s,d) = H(s,d)\, t_{\mathrm{r}} + T_{\mathrm{c}}(s,d) + \frac{L}{b} \tag{2}$$

As the load on the network increases, greater resource contention contributes to longer router traversal latencies.

A well designed network exploits available resources to improve performance. Simple, regular topologies provide compact layouts and allow low-latency router architectures to be used. However, routing latency begins to dominate as more processors are added and hop counts increase. While topologies using high radix routers can reduce the the average hop count, wiring requirements complicate layout and may increase the area consumed by the network. Similarly, increasing the channel bandwidth improves throughput at the expense of additional demand for wiring resources, which may complicate layout or dilate the die area. For a fixed wire bandwidth, increasing the channel width to reduce serialization latency improves performance but occupies more area. As the die area expands and packets travel longer distances, both channel latency and energy consumption increase.

2.2 Wire Model

We use the first-order RC wire delay model described in [8] and depicted in Figure 1. The model incorporates the following intrinsic parameters: the input or gate capacitance (C_g); the output or diffusion capacitance (C_d); the equivalent resistance (R); the resistance per millimeter for a minimum pitch wire (R_w); and the capacitance per millimeter for a minimum pitch wire (C_w). Parameter values appear in Tables 1 and 2.

Inverters are identically sized and uniformly distributed along the path. The repeated wire delay (T_w) across the segment of length l with repeaters of size K and PMOS:NMOS ratio ß is calculated as:

$$\kappa_0 = R(1+ß)(C_d + C_g) \tag{3}$$

$$\kappa_1 = \frac{R}{K}C_w + KR_w(1+ß)C_g \tag{4}$$

$$\kappa_2 = \frac{1}{2}R_wC_w \tag{5}$$

$$T_w = \kappa_0 + \kappa_1 l + \kappa_2 l^2 \tag{6}$$

All wires are minimum pitch with an aspect ratio consistent with our representative technology, which maximizes the aggregate channel bandwidth [14].

We calculate the power (P_w) dissipated driving the wire segment with activity factor α as:

$$\kappa_a = \alpha\left(K(C_g + C_d) + C_w l\right) f V_{\mathrm{DD}}^2 \tag{7}$$

$$\kappa_l = \frac{1}{2}K(I_{\mathrm{off_N}} + ßI_{\mathrm{off_P}})V_{\mathrm{DD}} \tag{8}$$

$$P_w = \kappa_a + \kappa_l \tag{9}$$

Here, $I_{\mathrm{off_N}}$ is the NMOS leakage current, and $I_{\mathrm{off_P}}$ is the PMOS leakage current.

Table 1: Interconnect Characteristics

Parameter	Local	Semi-global	Global
Metal Layers	4	2	2
Width (nm)	100	200	400
Pitch (nm)	200	400	800
Thickness (nm)	180	360	720
Aspect Ratio	1.8	1.8	1.8
Vertical Spacing (nm)	200	300	700
Resistivity (Ωmm)	2.8	2.5	2.3
Resistance (Ω/mm)	1550	350	80
Capacitance (fF/mm)	166	228	240

Table 2: Device Characteristics

Supply Voltage	V_{DD}	1.0	[V]
Gate Capacitance	C_g	1.34	[fF/um]
Diffusion Capacitance	C_d	0.85	[fF/um]
Equivalent Resistance	R	1.085	[kΩ/um]
NMOS Leakage Current	$I_{\mathrm{off_N}}$	30	[nA/um]
PMOS Leakage Current	$I_{\mathrm{off_P}}$	30	[nA/um]

2.3 Technology Model

Our technology model is intended to be representative of the 65nm processes disclosed in [2, 3, 13]. Device models are based on the predictive technology models described in [22]. Interconnect models were derived from [10, 2, 3, 13]. Tables 1 and 2 present relevant parameters.

3. SYSTEM ARCHITECTURE

In this section, we introduce the motivating tiled CMP architecture and identify the constraints imposed by the VLSI technology model used in the evaluation.

We consider a tiled CMP with 64 processors connected by a packet switched interconnection network implemented in a 65nm CMOS technology. Each tile includes an execution unit and 256KB of local memory. Because our focus is the interconnection network rather than on-chip coherence and consistency protocols, we model an abstract communication

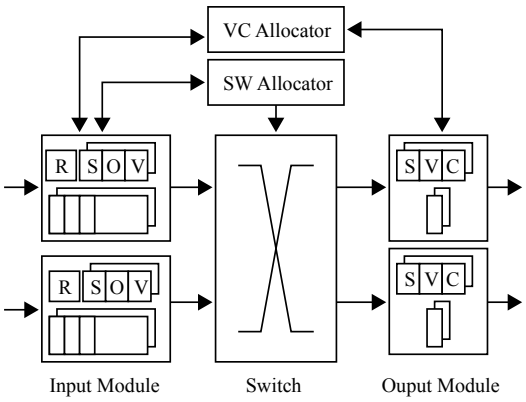


Figure 2: Router Architecture

protocol consisting of read and write transactions. Read transactions are initiated with a request sent to a remote tile and are completed when the requested memory block is received. Write transactions are initiated with a request that transfers a block to a remote tile and are completed when an acknowledgment from the remote tile is received. Transactions transfer 512-bit blocks of data. To model prefetching and similar latency hiding techniques, tiles may have up to four concurrent transactions pending. Read requests and write replies are mapped into short packets, which are nominally 64-bits in length. Read replies and write requests are mapped into longer packets that are constructed by appending the memory block to a header similar to the request packet, which yields a packet that is nominally 576-bits in length.

The processor tiles are nominally square, measuring 1.5mm along an edge. Local metal layers M1-M4 are reserved for the processor, leaving semi-global metal layers M5 and M6 available for routing network channels over the tiles. Global metal layers M7 and M8 are reserved for distributing power and ground, clocks, and critical control signals. The interconnection network is required to operate at the processor clock frequency of 2GHz, which corresponds to roughly 24 fan-out of 4 inverter delays. In-order packet delivery is not required.

4. NETWORK ARCHITECTURE

In this section we describe the network architecture and develop our area and energy models. We begin with a description of the router architecture in Section 4.1 and proceed to elaborate on the design of the datapath components in Sections 4.2 through 4.4. We describe the design of the channels in Section 4.5, and conclude with our area model for a complete router in Section 4.6.

4.1 Router Architecture

Figure 2 presents a block diagram of the pipelined input-queued virtual-channel router architecture used throughout this analysis. The architecture resembles that described in [18]. We use route lookahead to remove routing from the critical path. The virtual-channel allocator (VC Allocator) allows speculative assignment of output virtual-channels to packets arriving at the input modules. Similarly, the switch allocator (SW Allocator) permits speculative assignment of switch time slots to flow control digits (flits). The switch allocator prioritizes flits with non-speculative virtual-channel assignments ahead of speculative requests. With route lookahead and speculation, the nominal router traversal time is 2 cycles. Routing, virtual-channel allocation, and switch assignment are performed in the first cycle, with switch traversal occurring in the second.

The router architecture often differentiates between short packets (read requests, write acknowledgments) and long packets (read responses, write requests). We denote the width of flits used when routing short packets as w_s, and denote the width of flits used when routing long packets as w_l. Short packets are mapped into a single flit, while long packets are mapped into a sequence of L_l flits. Thus, for long packets we have: packet length $= w_l \times L_l$. The datapath width (w) corresponds to the maximum width of flits traversing the network: $w = \max(w_s, w_l)$.

4.2 Input Module

The input module manages the states of its virtual-channels and buffers flits until they can be forwarded. The following state, illustrated in Figure 2, is replicated per virtual-channel: the virtual-channel control state (S), which indicates whether the virtual-channel is in use; the destination output module (O); and the assigned output virtual-channel (V). We do not reuse a virtual-channel until the tail credit is received. Consequently, it is not necessary to track the number of credits available for the virtual-channel at the downstream router. Instead, the virtual-channel is held until the downstream router signals that the tail flit has been forwarded, freeing the virtual-channel for recycling. A routing unit (R) is implemented in each input module. With route lookahead, the routing unit simply decodes the routing information provided by the previous router to determine the output virtual-channel for a packet.

Flits arrive late in the clock cycle and are retimed by an input register before being buffered. The flit buffers are implemented in a dual-port memory. Each virtual-channel receives a disjoint allocation of contiguous memory words to allow rapid translation of virtual-channel identifiers to buffer locations. The flit buffers are bypassed when an arriving flit is immediately eligible for virtual-channel and switch allocation. Flits are written into the buffers regardless of whether the bypass path is enabled to allow recovery from a failed speculative allocation.

The memory width is matched to the datapath width to allow an entire flit to be accessed in one cycle. Because short flits can be accommodated by a narrower memory word, the memory includes two SRAM arrays of different widths. The arrangement of the arrays within the memory is illustrated in Figure 3 (which is described in more detail below). This construction improves the energy efficiency of the flit buffer and is possible because we statically assign virtual-channels locations in the memory. To avoid unnecessary transitions at the egress of the input module when forwarding short flits to the switch, the flit buffer includes a set of latches (QLatch) to preserve a quescient state on lines that are not used by short flits.

4.2.1 Input Module Area Model

The area of the input module is dominated by the flit buffer memory. Figure 3 illustrates the placement of blocks used to estimate the area of the input module. The datapath uses a bitslice construction, with the slice pitch selected to

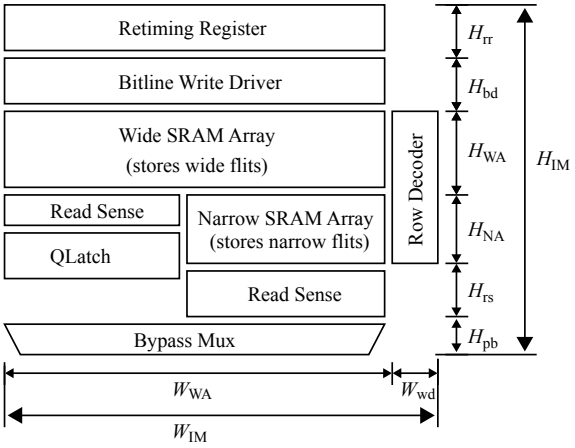


Figure 3: Router Input Buffer Area

accommodate both the memory cell pitch and and the switch input pitch. Arrays of elements whose pitch exceeds the slice pitch, such as the latches, are folded to preserve a constant pitch through the datapath pitch.

Because narrow memories result in more compact layouts for our floorplans, we assume no bitline multiplexing at the sense amps. For an SRAM cell with dimensions $H_{\text{cell}} \times W_{\text{cell}}$, the height of the wide array (H_{WA}) is determined by the number of virtual-channels provided for long packets (VC_l) and the buffer depth per virtual-channel (D_l):

$$H_{\text{WA}} = H_{\text{cell}} \times VC_l \times D_l \tag{10}$$

The height of the narrow array is similarly determined by the number of virtual-channels provided for short packets (VC_s) and the buffer depth per virtual-channel ($D_s = 1$) .

$$H_{\text{NA}} = H_{\text{cell}} \times VC_s \times D_s \tag{11}$$

The height of the bitline drivers (H_{bd}) and the read sense logic (H_{rs}) depends on the circuit topology. When detailed circuit designs are unavailable, we estimate the height of the bitline drivers based on the size of inverters needed to drive the bitlines and the area required for the bitline conditioning circuits. We estimate the height of the read sense logic assuming a cross-coupled inverter pair construction. The height of the flit buffer memory (H_{Mem}) is

$$H_{\text{Mem}} = H_{\text{WA}} + H_{\text{NA}} + H_{\text{bd}} + H_{\text{rs}} \tag{12}$$

When in the height of the retiming register (H_{rr}) and bypass multiplexer (H_{bp}) are included, the total height of the input module (H_{IM}) is:

$$H_{\text{IM}} = H_{\text{rr}} + H_{\text{Mem}} + H_{\text{bp}} \tag{13}$$

The width of the input module (W_{IM}) is determined by the width of the wide array (W_{WA}) and the row decoder driving the wordlines (W_{wd}):

$$W_{\text{IM}} = W_{\text{IM}} + W_{\text{wd}} = (w \times W_{\text{cell}}) + W_{\text{wd}} \tag{14}$$

The decoder width is estimated from the size of the final buffers required to drive the wordlines and the number of predecode wires routed along the length of the row decoder, parallel to the array bitlines. As will be illustrated in Figure 7, the placement of the row decoder allows its width to be partially hidden by the output modules described subsequently.

4.2.2 Input Module Energy Model

The wordline capacitance (C_{wl}) is calculated from the number of bits (N_{b}) per memory word, the gate capacitance of the passgate transistor (C_{pg}), and the wire capacitance associated with the length of the wordline spanning a cell (l_w). For the wide array $N_{\text{b}} = w_l$, while for the narrow array $N_{\text{b}} = w_s$. The wordline capacitance is calculated as:

$$C_{\text{wl}} = N_{\text{b}}(2C_{\text{pg}} + C_w l_w) \tag{15}$$

The bitline capacitance (C_{bl}) is calculated from the number of words in the array (N_{w}), the diffusion capacitance of the passgate transistor (C_{pd}), and the wire capacitance of the length of the bitline spanning the cell (l_b):

$$C_{\text{bl}} = 2N_{\text{w}}(C_{\text{pd}} + C_w l_b) \tag{16}$$

The energy expended in a write operation is calculated based on the discharging of the bitlines by the bitline driver with internal capacitance (C_{bd}) and the charging of the wordline by the wordline driver with internal capacitance (C_{wd}). The energy expended activating a wordline (E_{wl}) is:

$$E_{\text{wl}} = (C_{\text{wd}} + C_{\text{wl}})V_{\text{DD}}^2 \tag{17}$$

We include the internal capacitance of the input retiming register (C_{rr}) in the write energy because the register is always written in the cycle before the flit buffer. For modeling purposes, we assume that half of the enabled cells change state during a write. The state change requires charging the internal capacitance of the cross-coupled inverters (C_{cc}). Accordingly, the energy expended writing a flit of width w_f into the flit buffer (E_{wr}) is:

$$E_{\text{wr}} = E_{\text{wl}} + w_f(C_{\text{rr}} + C_{\text{bd}} + C_{\text{bl}} + 0.5\,C_{\text{cc}})V_{\text{DD}}^2 \tag{18}$$

The energy expended in a read operation is calculated similarly to that of a write operation except that we assume the bitlines have a signal swing of $0.25V_{\text{DD}}$ during a read operation. For modeling purposes, the capacitance associated with the read sense circuit (C_{rs}) absorbs that associated with the buffers required to drive the signals to the switch. The energy expended by a read of the flit buffer (E_{rd}) is:

$$E_{\text{rd}} = E_{\text{wl}} + C_{\text{rs}}V_{\text{DD}}^2 + 0.25C_{\text{bl}}V_{\text{DD}}^2 \tag{19}$$

4.3 Switch

The switch is implemented as a segmented crossbar to allow a regular, compact layout and reduce power dissipation [21]. Figure 4 presents a simplified illustration of a 2-input, 2-output segmented crossbar. As suggested by Figure 4, the segmented crossbar is a refinement of the canonical matrix crossbar in which the input and output lines are broken into segments of approximately equal length with adjacent segments isolated by tri-state buffers. Control signals selectively enable the tri-state buffers to minimize the number of active line segments. All of our routers use segmented crossbars that divide the input and output lines into two segments of approximately equal length.

4.3.1 Switch Area Model

Each edge of the crossbar must be wide enough to accommodate $N_{\text{i}} \times w$ input signals or $N_{\text{o}} \times w$ output signals. The crossbars considered in this analysis are wire dominated,

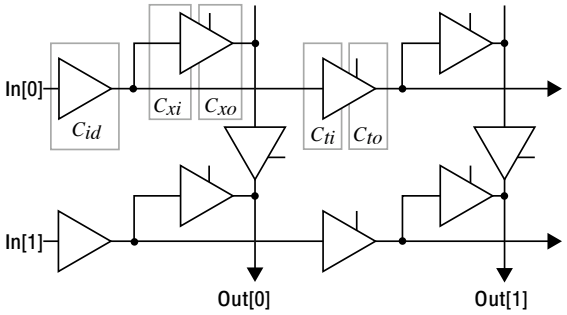


Figure 4: Crossbar annotated with Capacitances

with areas proportional to $(N_\mathrm{i} \times w) \times (N_\mathrm{o} \times w)$. When calculating the dimensions of a crossbar, we must account for metal tracks reserved for vias bringing signals from the upper wire matrix layers down to the crosspoints. We denote the spacing between wires as s_x. For a wire pitch of p_w, the crossbar width and height are calculated as:

$$W_x = N_\mathrm{o} \times (w \times p_w \times s_x) \quad (20)$$

$$H_x = N_\mathrm{i} \times (w \times p_w \times s_x) \quad (21)$$

4.3.2 Switch Energy Model

The model for the capacitive load contributed by the input lines incorporates the following parameters: the length of the input wire (l_{xb}); the capacitance contributed by the input driver (C_{id}); the capacitive contributed by the input of a cross-point (C_{xi}); the input capacitance of the tri-state buffer (C_{ti}); the output capacitance of the tri-state buffer (C_{to}); and, the wire capacitance per unit length (C_w). When only the first segment of an input line is driven, the load is calculated as:

$$C_{xbi} = C_{id} + 0.5(N_o C_{xo} + C_w l_{xb}) + C_{ti} \quad (22)$$

When both segments of an input line are driven, the load is calculated as:

$$C_{xbi} = C_{id} + N_o C_{xi} + C_w l_{xb} + C_{ti} + C_{to} \quad (23)$$

The model for the capacitive load contributed by the output lines incorporate the following additional parameters: the parasitic diffusion capacitance at the output of a crosspoint switch (C_{xo}); and the capacitance associated with the output latch (C_l). When only the first segment of an output line is driven, the load is calculated as:

$$C_{xbo} = 0.5(N_i C_{xo} + C_w l_{xb}) + C_{to} + C_l \quad (24)$$

When both output segments are driven, the output load is calculated as:

$$C_{xbo} = N_i C_{xo} + C_w l_{xb} + C_{ti} + C_{to} + C_l \quad (25)$$

The energy associated with a flit of width w_f traversing the switch can be calculated as:

$$E_\mathrm{sw} = w_f \times (C_{xbi} + C_{xbo}) V_\mathrm{DD}^2 \quad (26)$$

4.4 Output Module

The output module forwards flits from the switch to the downstream router. Because traversing the switch consumes enough of the clock period to prohibit propagating the flit to the downstream router in what remains of the cycle, it is necessary to latch outgoing flits in the output module. To reduce energy consumption when sending flits that are narrower than the channel, the output latch preserves a quiescent state on those wires that are not used to transmit the flit. The state fields of interest are: the output virtual-channel control state (S), the input virtual-channel (V), and the number of flit buffer credits available at the downstream router (C). The control state indicates whether a virtual-channel is idle, active, or unavailable because the tail credit of the previous packet has yet to be returned.

4.4.1 Output Module Area Model

The output latch contributes most of the output module's area. Accordingly, we use the dimensions of the output latch to determine the height of the output module (H_OM) and the width of the output latch to determine the width of the output module (W_OM).

4.4.2 Output Module Energy Model

We limit the energy consumption attributed to the output module to that of the output latch not included in the crossbar and incorporate the energy expended switching the channel wires into the channel energy model. For a datapath of width w, the energy dissipated when forwarding a flit of width w_f through the output module (E_OM) is:

$$E_\mathrm{OM} = w_f E_\mathrm{L} + w C_{\mathrm{L},in} V_\mathrm{DD}^2 \quad (27)$$

Here, E_L is the energy associated with an active latch transition and $C_{\mathrm{L},in}$ is the input latch capacitance loading a crossbar output. The later term correctly accounts for switched capacitance at crossbar outputs leading to disabled latches when forwarding short flits.

4.5 Channels

Figure 5 illustrate how the number (N) and size (K) of repeaters affects the distance a signal routed in semi-global metal layers will propagate in one cycle and the energy expended driving the repeated wire. At shorter wire lengths, the wire contributes 10 to 40 times the capacitance of the repeaters. At longer wire lengths where repeaters become larger and more frequent, the contribution of the wire capacitance declines to 6 to 10 times that of the repeaters. The similarity of the curves presented in Figure 5 results from the majority of the energy being expended in the wires.

Wires transporting signals traveling longer distances are broken into M segments separated by sequencing elements. With channel lengths determined by the placement of routers in the floorplan, we design channels by selecting K, N, and M to minimize the energy-delay measure KNM^2 subject to the constraint that the repeated segment wire delay (T_w) cannot exceed the cycle time less margin for setup time and clock skew.

4.5.1 Channel Area Model

The lateral capacitance between wires running parallel over long distances results in capacitive coupling which affects timing and leads to coupling noise injection. To mitigate these adverse effects, metal tracks are reserved at regular intervals within the channels for power and ground shields. We calculate the physical width of the region reserved for routing a channel (W_ch) from the width of the channel (w), the metal pitch (p_M) and the average signal

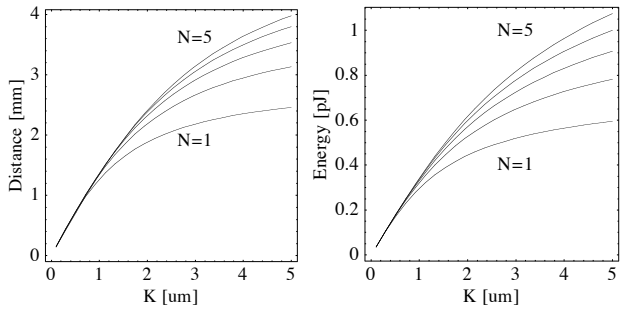


Figure 5: Channel Wire Performance

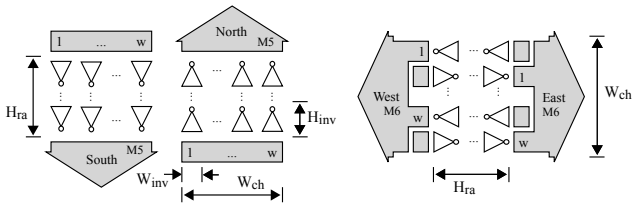


Figure 6: Channel Area Calculation

wire spacing within the channel (s_w):

$$W_{\text{ch}} = w \times p_M \times s_w \tag{28}$$

We reserve area within the processor tiles for repeaters as necessary and increase the dimensions of the tiles appropriately. When allocating area for a collection of repeaters within a processor tile, we reserve a rectangular strip of uniform height across the tile. If necessary, we fold the repeater array maintain the signal wire pitch at the expense of an increase in the array's height. Folding causes the height of the reserved region to increase as signal densities increase in wider channels. Figure 6 illustrates the folding of repeaters. As illustrated in the right of Figure 6, wires from parallel channels may be interdigitated to allow a compact layout while maintaining the datapath pitch established in the routers. For an inverter cell of height H_{inv} and width W_{inv}, we calculate the height of the repeater array (H_{ra}) using the channel width from (28) as:

$$H_{\text{ra}} = H_{\text{inv}} \lceil w W_{\text{inv}} / W_{\text{ch}} \rceil \tag{29}$$

4.5.2 Channel Energy Model

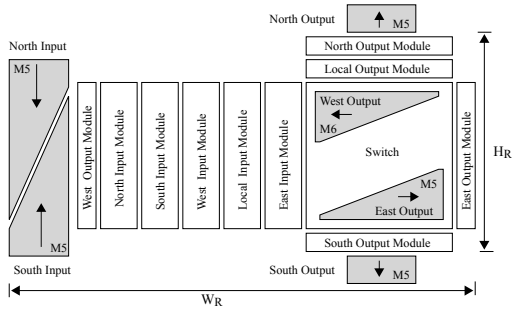
The channel energy model is derived from the wire model of Section 2.2. The energy dissipated in a w-bit wide channel with M sequencing elements and N repeaters per segment is:

$$E_{\text{ch}} = wM(E_{\text{sq}} + NE_{\text{w}}) \tag{30}$$

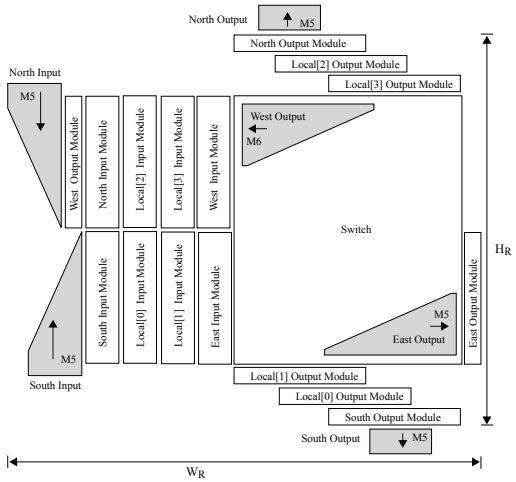
Here, the energy expended driving the wire (E_{w}) is calculated from (7), and the energy dissipated in the sequencing element (E_{sq}) is based on an analytic energy model of a hybrid-latch flip-flop we developed and verified with SPICE simulations.

4.6 Combined Router Area Model

The router area is determined by the placement of the datapath elements. For illustrative purposes, we present layouts for two router architectures used in our analysis. Figure



(a) 5×5 Router



(b) 8×8 Router

Figure 7: Router Floorplan for Area Model

7(a) illustrates the arrangement of datapath elements in a 5-input, 5-output router suitable for a mesh or torus network. The input modules are collocated to one side of the switch to reduce the length of control signals traveling between the input modules and the arbiters. Vertical channels connecting to the North and South neighbors are implemented in M5, while horizontal channels connecting to the East and West neighbors are implemented in M6. Inputs from the North and South neighbors are brought down to the local metal layers at the left edge of the router before traveling horizontally to an input module. Inputs from the East and West neighbors are brought down from M6 over the destination input module. Outputs to the North and South neighbor leave from the top and bottom edges of the switch, respectively. Because output lines in the switch run vertically, the East and West neighbor outputs are brought up to the semi-global metal layers over the switch before traveling horizontally to the output modules. The connection from the switch to the West output module is implemented in M6; the required vias create a blockage in M5 that precludes routing vertical channels over the switch. The connection from the switch to the East output module is implemented in M5, which departs from our convention of routing only vertical signals in M5. However, there are too few tracks available in M6 to route both outputs and the East input channel over the switch in a single metal layer. The width (W_{R}) and height (H_{R}) of the 5×5 router depicted in Figure 7(a) are

calculated as:

$$W_{\mathrm{R}} = W_{\mathrm{ch}} + 5H_{\mathrm{IM}} + 2H_{\mathrm{OM}} + W_x \quad (31)$$
$$H_{\mathrm{R}} = 3H_{\mathrm{OM}} + H_x \quad (32)$$

Figure 7(b) illustrates organization of an 8-input, 8-output router suitable for an indirect network. The increased switch height allows the input modules to be stacked vertically to reduce the width of the router while maintaining a consistent slice pitch across the router. The width (W_{R}) and height (H_{R}) of the 8×8 are calculated as:

$$W_{\mathrm{R}} = W_{\mathrm{ch}} + 4H_{\mathrm{IM}} + 2H_{\mathrm{OM}} + W_x \quad (33)$$
$$H_{\mathrm{R}} = 6H_{\mathrm{OM}} + H_x \quad (34)$$

5. NETWORKS

In this section we describe the networks evaluated in this analysis. The network topologies are illustrated in Figure 8. Figure 9 illustrates the placement of routers amongst the processor tiles in the lower left quadrant of the chips used to estimate chip areas. Channels connecting routers outside the quadrant are omitted for clarity.

5.1 Mesh/MeshX2

Topology: The mesh topology is depicted in Figure 8(a). The Mesh network leaves enough unused area between the tiles to implement a second network. The MeshX2 network increase wire and area utilization by introducing a second parallel network.

Routing: We use the O1TURN algorithm described in [19]. The algorithm is a randomized variant of dimension-order routing in which the order of dimension traversal, XY or YX, is chosen at random to evenly distribute the load over the XY and YX paths. Each subnetwork of the MeshX2 architecture transports a disjoint subset of the packet types. Assigning short packets to one subnetwork and long packets to the other reduces the complexity of the subnetworks because each transports packets of a single, fixed length. Alternately, assigning read request and response packets to one subnetwork and write request and response packets to the other, effectively creating a read subnetwork and a write subnetwork, allows a single network architecture to be used for both subnetworks. We shall show that partitioning the networks based on the packet lengths performs poorly when such a partitioning fails to maximize use of the available wiring resources.

Flow Control: Deadlock is avoided by introducing a deadlock free routing subfunction accessible through a distinguished subset of the virtual-channels [6]. We reserve one virtual-channel for XY dimension-order routing and one virtual-channel for YX dimension-order routing. The remaining virtual-channels are available for both XY and YX routing.

5.2 Torus

Topology: The Torus is constructed from the Mesh by adding end-around channels at the periphery. We use the folded topology depicted in Figure 8(b), which provides channels of uniform length at the expense of longer average channel lengths.

Routing: The Torus network uses dimension-order routing.

Flow Control: Intra-dimension deadlock is avoided by partitioning each ring and promoting packets into reserved sets of virtual channels when they cross the partition [5].

5.3 Concentrated Mesh (CMesh/CMeshX2)

Topology: The CMesh network reduces the Mesh described above to a radix-4 mesh in which each router services four processors. The decrease in the average hop count reduces the component of the zero-load latency due to router delay (T_{r}). We augment the mesh with express channels along the perimeter of the network to restore the bisection channel count lost to the reduction in the radix of the mesh. As with the Mesh networks, we can improve the wire and area utilization of the CMesh network by introducing a second parallel network. The CMeshX2 network provides two independent CMesh subnetworks.

Routing: The CMesh network uses dimension-order routing with the express links used when a packet travels more than one hop along the perimeter of the network. While we could conceivably reduce the average number of hops packets travels by preferentially routing over the express channels even when doing so violates the normal dimension order, such a routing strategy would introduce deadlock and is therefor disallowed.

Flow Control: The routing algorithm is deadlock free and therefor introduces no additional constraints on the flow control policy.

5.4 Fat-Tree (FTree)

Topology: The FTree topology is based on that described in [12]. As illustrated in Figure 8(d), we use a 4-ary fat-tree with a height of 3, which corresponds to an instance of the SPIN network described in [1]. Like the tapered fat-tree described below, the FTree is also an instance of a folded Clos network.

Routing: The FTree topology uses nearest-common ancestor routing. Packets are adaptively routed up to the common ancestor and deterministically down to the destination.

Flow Control: The fat-tree does not introduce addition restrictions on the allocation of virtual-channels beyond those required to prevent protocol deadlock.

5.5 Tapered Fat-Tree (TTree)

Topology: The TTree, illustrated in Figure 8(e), is an instance of a generalized fat-tree topology [16] in which the bandwidth decreases towards the root. The topology is equivalent to a tapered 5-stage folded Clos with reduced bandwidth in the interior of the network and interior nodes replicated to increase path diversity.

Routing: The routing algorithm is similar to that used for the FTree. Should a packet reach a root node while routing up the tree, the algorithm randomly forwards the packet to one of the two viable descendents.

Flow Control: The routing algorithm for the TTree network does not introduce any additional constraints on the allocation of virtual channels beyond those required to avoid protocol deadlock.

5.6 Network Comparison

Table 3 compares the network configurations that provided the best area and energy efficiencies. We identified these configurations by evaluating the impact of the following design parameters on the performance and efficiency: the number of virtual-channels assigned to each packet type and the allocation policy; the sharing of virtual-channels across

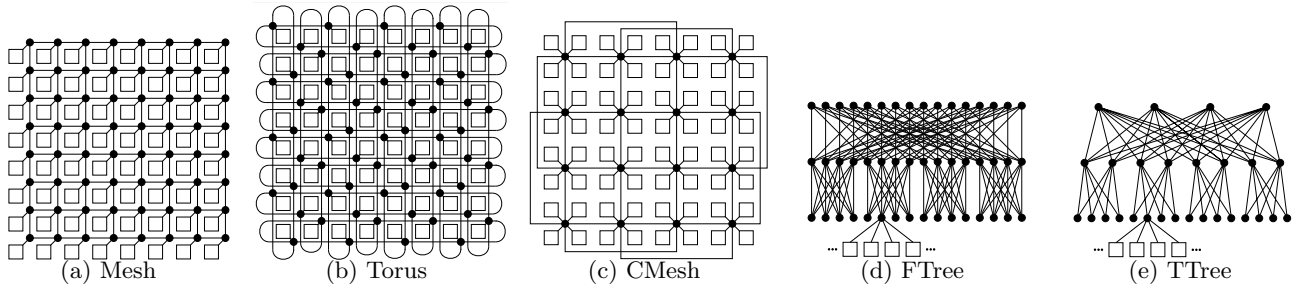


Figure 8: Network Topologies

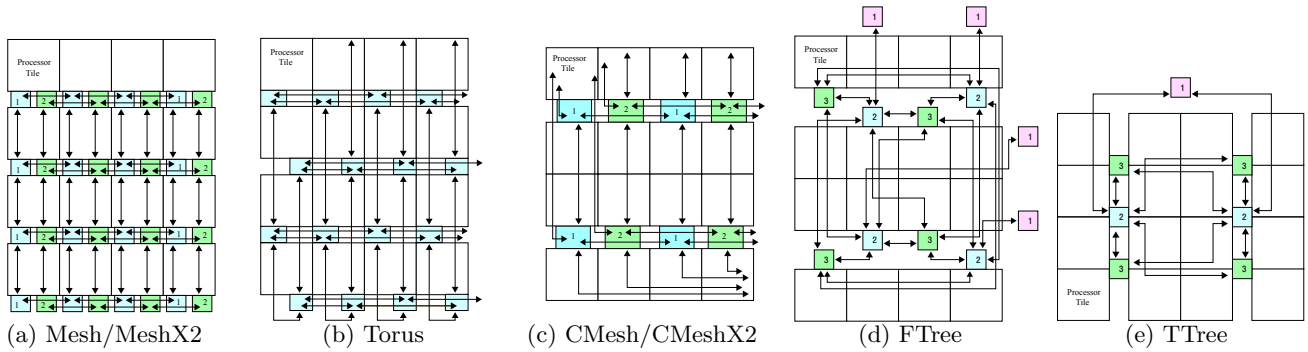


Figure 9: Floorplan for Estimating Area (Lower Left Quadrant of Die Shown)

packet types; the flit buffer depth provided for each packet type; the channel router datapath width; the assignment of packet types to subnetworks; the routing function; and the channel repeater sizing and spacing policy. The large switch used in the CMesh configurations introduces an additional router stage for switch preparation, which increases the router latency to 3 cycles. Table 4 lists the number of virtual-channels allocated for each packet type and the flit buffer depth per virtual-channel.

Table 3: Preferred Network Configurations

	H	t_r	B_C	w	B_B	T_c	T_s	T_0
Mesh	$6\frac{1}{4}$	2	16	192	3,072	5.3	3	17.8
MeshX2	$6\frac{1}{4}$	2	32	192	6,144	5.3	3	17.8
Torus	5	2	32	288	9,216	4.0	2	14.0
CMesh	$3\frac{1}{8}$	3	16	288	4,608	2.1	2	11.5
CMeshX2	$3\frac{1}{8}$	3	32	288	9,216	2.1	2	11.5
FTree	$4\frac{3}{8}$	2	64	144	9,216	4.4	4	13.1
TTree	$4\frac{3}{8}$	2	32	144	4,608	3.5	4	12.2

6. EVALUATION METHODOLOGY

We evaluate the networks based on their area and energy efficiency using a cycle-accurate interconnection network simulator that incorporates our area and energy models. Area efficiency is measured as the product of the workload completion time and the chip area; energy efficiency is measured as the product of the completion time and energy expended in the network. For each network, we evaluate the impact of varying the datapath width, number and partitioning of virtual-channels, flit-buffer depths, and placement of routers amongst the tiles to determine the configuration providing the most competitive area and energy efficiency. To accurately estimate energy dissipation in different configurations, our simulator automatically sizes repeaters based on the calculated channel lengths, accounts for the impact of the flit-buffer sizing on the memory circuitry, and adjusts driver sizes within the router datapath to match capacitive loads calculated using our energy models. The area model parameters used in the evaluation are presented in Table 5.

Table 4: Virtual-Channel Configurations

	Short Packet		Long Packet	
	Number	Depth	Number	Depth
Mesh(X2)	8	1	6	3
Torus	8	1	6	2
CMesh(X2)	8	1	8	2
FTree	8	1	4	4
TTree	8	1	4	4

We simulate a workload in which every processor performs a fixed number of transactions and measure the time required for all transactions to complete. We follow the convention of ensuring the number of messages exchanged during the simulation is large enough that those packets injected into the empty network at startup, which experience little queuing delay and therefore lower latency, do not unduly influence the reported results. Each processor issues at least 1,000 packets, which was determined to be sufficient to ensure that load-induced queuing delay develops.

Table 5: Area Model Parameter Values

Parameter	Value (M1 Tracks)
SRAM Cell Height (H_{cell})	8
SRAM Cell Width (W_{cell})	6
Latch Height	10
Latch Width	10
Read Sense Height (H_{rs})	40
Bitline Driver Height (H_{bd})	20
Inverter Height (H_{inv})	$10 + 3K/5$
Crossbar Wire Spacing (s_x)	2
Channel Wire Spacing (s_w)	2

The mixture of common synthetic traffic patterns included in the workload (bitreverse, nearest neighbor, tornado, and uniform random) was chosen to avoid biasing the workload in favor of one topology or routing algorithm. The taper pattern corresponds to an augmented uniform random pattern in which the probability of nodes communicating decreases with the distance between them. Rather than report throughput-latency curves, we simulate a closed-loop system in which each sender and receiver may have a limited number of messages in-flight, which is more reflective of a CMP. Aggregate workload completion times are reported because they better reflect the network's overall performance than the completion time of any single traffic pattern or trace.

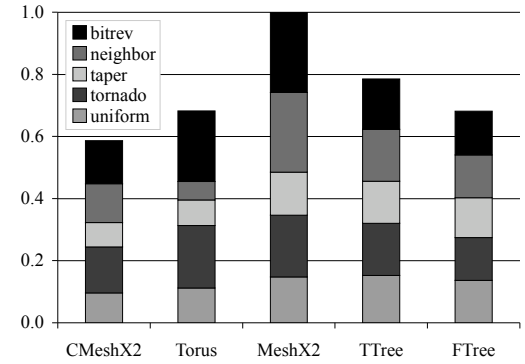
7. RESULTS AND ANALYSIS

This section presents the results of our analysis. We contrast the performance of the networks described above to provide insight into the merits of the different architectures. Figure 10 compares the performance and efficiency of the preferred configuration of each network identified in Table 3. Figure 10(a) presents the relative workload completion times. Figure 10(b) separates chip area estimates into major components. Figure 10(c) contrasts the average power dissipation in the networks during the simulated workload. Normalized area-delay and energy-delay metrics are presented Figure 10(d). We discuss the preferred topology, network costs, benefits of a second network, and load balancing issues in the following subsections.

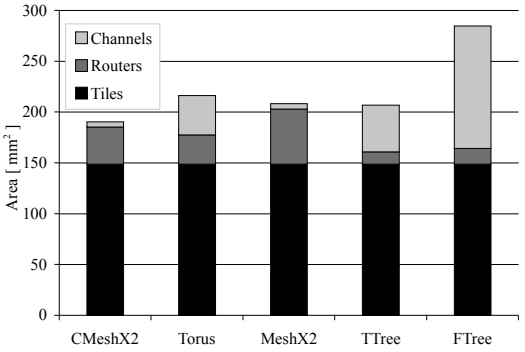
7.1 Preferred Topology

As Figure 10(a) illustrates, the CMeshX2 network provides the best aggregate completion time. Additionally, the CMeshX2 network provides the best completion time for uniform and tapered traffic patterns, which one might consider more representative of typical application behavior. Figure 11, which illustrates the distribution of packet latencies for the uniform pattern component of the workload, shows that the CMeshX2 network reduces the latency variability in addition to providing a lower average latency. While the distributions exhibit lengthy tails, indicating occasional long packet latencies, the networks are heavily loaded during the workload and operate near capacity.

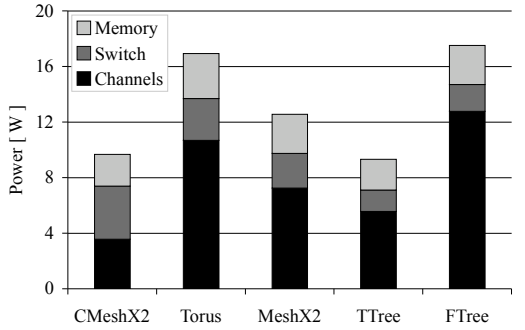
Figure 10(d), which compares network efficiency, clearly shows that the CMeshX2 network is substantially more efficient. Concentration reduces the average hop count and improves load balance across the channels without increasing wiring complexity. While the tree networks also offer low hop counts, their wiring complexity limits the datapath



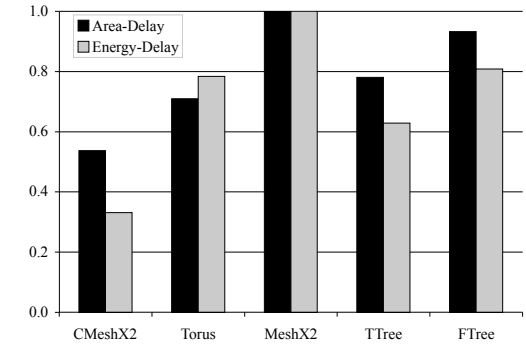
(a) Relative Workload Completion Time



(b) Chip Area

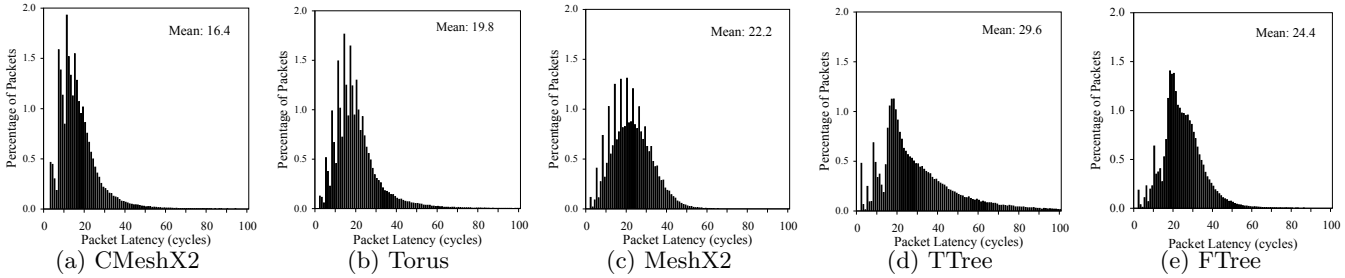


(c) Network Power Dissipation



(d) Relative Efficiency

Figure 10: Performance Comparison



(a) CMeshX2 (b) Torus (c) MeshX2 (d) TTree (e) FTree

Figure 11: Workload Packet Latency Distribution for Uniform Random Traffic Pattern

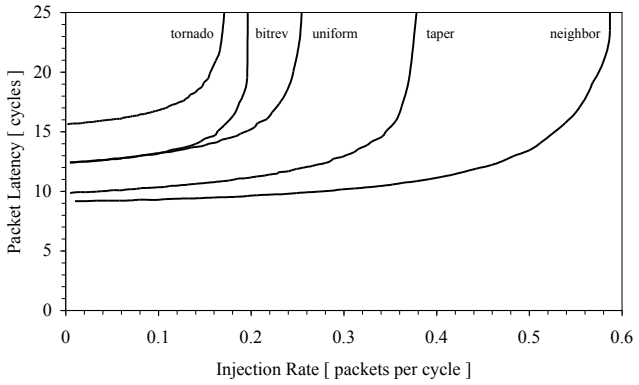
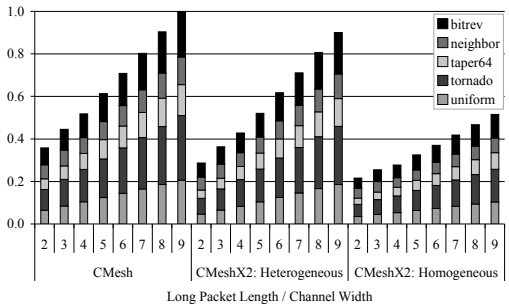
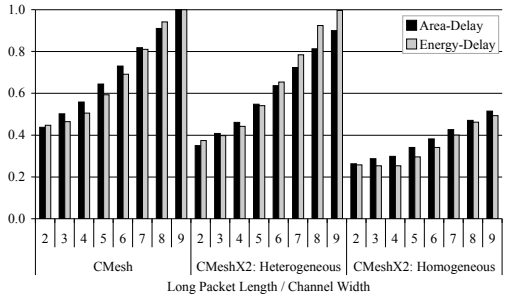


Figure 12: Offered Latency for CMeshX2 Network



(a) Relative Workload Completion Time



(b) Relative Efficiency

Figure 13: CMesh Performance

width and results in long channels with significant traversal costs. The power dissipated in the channels relative to the bisection bandwidth is substantially greater in the tree networks as a consequence of the wiring complexity and ensuing area increase. The Torus network similarly suffers from excess energy dissipation in the channels. The Torus offers a compact layout but its average channel length is long, spanning two tiles and the bypassed router in between. The repeaters used to drive these long channels consume more energy per length of wire driven than those used in the CMeshX2 network, resulting in worse energy efficiency. The mesh networks, which are commonly proposed architectures for on-chip networks, perform poorly and offer the worst energy and area efficiency of all networks evaluated in this study.

Figure 12 plots measured latencies against packet injection rate for the CMeshX2 network for different traffic patterns. Measurements were taken using a uniform distribution of packet types. The curves effectively illustrate the network's ability to exploit locality in traffic flows to reduce latency.

7.2 Network Costs

Figure 10(b) illustrates the contributions of the processor tiles and network to the total die area. The channel component includes area allocated for routing channels over what might otherwise be unused die area. While the unused lower metal layers and diffusion area could be partially reclaimed for other purposes, the regions tend to be irregularly shaped with frequent interruption by repeater arrays and are consequently unsuitable for larger structures such as memories.

Figure 10(b) shows that the direct topologies and sparse indirect topology incur similar area costs. However, the direct topologies afford wider datapaths by routing more channels above processor tiles rather, which avoids dedicating additional area to routing channels around the die. The regular planar structure of the direct networks renders them particularly amenable to compact layout and well suited to the inherent structure of a tiled CMP. The area consumed by the CMesh network is modest, contributing at most 23.6% of the total die area with two parallel 288-bit datapaths to as little as 8.4% with two parallel 64-bit datapath.

Figure 10(c) compares the power consumed by the networks, showing the contributions from the different datapath components. The power consumed in the networks is modest and would represent an acceptable component of the total power dissipated by the chip. While the TTree network dissipate slightly less power than the CMeshX2 network, the later is simply more active and Figure 10(d) clearly illustrates that the CMeshX2 network is more energy efficient.

This efficiency results from the lower hop count, which reduces the energy expended routing packets, and more compact layout, which reduces channel lengths and the energy dissipated driving wire capacitance in the channels.

7.3 Benefits of a Second Network

Introducing a second parallel network improves utilization of the abundant wire resources. While increasing the datapath width would also improve wire utilization, the larger crossbars required in wider datapaths consume more die area and dissipate more energy when forwarding packets. To quantify the benefits of a second network, we compare a single concentrated mesh network to duplicated configurations. We consider two strategies for distributing traffic over the subnetworks: the heterogeneous architecture, and the homogeneous architecture. The heterogeneous architecture uses one subnetwork to transport short packets (read requests, write replies) and the other to transport long packets (read replies, write requests). The homogeneous architecture uses one subnetwork to transport packets associated with read transactions (requests and replies) and one to transport packets associated with write transactions.

Figure 13 compares the performance of a single concentrated mesh network to the replicated architectures. We vary the datapath width to illustrate its effect on the network's performance. The widths in Figure 13 yield serialization latencies for long packets of 2 to 9 cycles. The datapath width of the short packet network in the heterogeneous architecture remains constant at 64-bits.

The addition of a second network significantly improves both performance and efficiency. The second network has negligible impact on the chip area because the additional routers reside in areas initially allocated for channels in the first network. While the presence of additional routers and channels increases power dissipation in the network, the greatly improved performance more than compensates, resulting in significant improvements in energy-efficiency.

While the performance of the two architectures is similar with wide datapaths, the homogeneous architecture consistently performs better. As Figure 13 illustrates, the performance advantage increases with narrower datapaths. For a wide datapath, the difference in performance results from the greater aggregate bandwidth provided by the homogeneous architecture. However, the homogeneous architecture performs significantly better even when both subnetworks provide identical 64-bit datapaths and equivalent aggregate bandwidth. The improvement arises from differences in how the two strategies distribute traffic over the subnetworks. The heterogeneous strategy routes all long packets through one subnetwork. Because the communication protocol generates short and long packets in equal numbers, the load on the long packet subnetwork is significantly greater and the short packet subnetwork is poorly utilized. Thus, the performance difference arises from the failure of the heterogeneous architecture to balance the load equitably across the subnetworks.

7.4 Contribution from Express Channels

To quantify the performance contributed by the express channels, we present the performance of the CMesh network when the express channels are removed. Table 6 compares the area and energy efficiency measured for the ensemble workload when the express channels are removed. With dimension-order routing (DOR), the workload completion time is reduced by 23.1%. Without express channels the area efficiency degrades by 23% and the energy efficiency degrades by 38%. Because the express channels are routed over processor tiles in otherwise unused metal tracks and use preexisting router ports, the area overhead is negligible. Energy efficiency improves both because the completion time decreases and because it is more efficient to route packets over express channels than through intermediate routers.

Table 6: Contribution of Express Channels

Express	Routing	Area-Delay	Energy-Delay
Present	O1TURN	0.800	0.713
	DOR	0.801	0.721
Absent	O1TURN	0.993	0.995
	DOR	1.000	1.000

Table 7: Permutation Completion Time

Network	Max	Min	Mean	Std.Dev
CMeshX2	0.649	0.600	0.689	0.007
Torus	0.903	0.876	0.890	0.004
MeshX2	1.000	0.928	0.991	0.002
FTree	0.662	0.641	0.649	0.003
TTree	0.949	0.890	0.924	0.009

7.5 Load Balancing

Non-deterministic routing can improve load balance. The CMeshX2 network performs equally well with deterministic and oblivious routing, illustrated by the similar performance of DOR and O1TURN (Table 6). The comparable performance is noteworthy because the workload heavily loads the networks and oblivious routing generally provides better throughput under heavy load. Because deterministic routing preserves message ordering between a source and destination, which reduces the complexity of cache coherence and similar protocols, we prefer DOR on the CMesh networks.

Table 7 presents the mean workload completion times for $5,000$ random traffic permutations. While the FTree network provides a slight performance improvement over the CMeshX2 network, the advantage does not compensate for the increased overhead. The poor performance of the Torus network on the random permutations in comparison to its performance on synthetic patterns results from the Torus being better able to exploit locality in the neighbor and taper patterns (similar results were obtained using adaptive routing strategies on the Torus network).

8. CONCLUSION

In this work we present detailed analytical area and energy models for on-chip networks and explore tradeoffs in the design of high performance interconnects for CMPs. Our analysis shows that architectures commonly assumed in on-chip networks studies are unlikely to perform well in CMP systems as processor counts increase. Even with low-latency router architectures assumed, a two dimensional mesh scales poorly in an on-chip environment.

We address the issue of channel width selection and demonstrate that increasing the channel width in on-chip networks

significantly improves both performance and area and energy efficiency. The serialization latency may contribute a significant component of the overall latency in an on-chip network, particularly when traffic is well localized or the network is lightly loaded. Wide channels exploit the availability of wires on-chip to provide significant improvements in the performance for packet sizes typical for CMP systems. The additional wiring complexity introduced by wide channels increases the importance of selecting a topology whose structure is well suited to the wiring constraints imposed on-chip. The indirect topologies evaluated in this work do not provide compact, area efficient layouts with wide channels, which adversely constrains the maximum channel width.

We demonstrate that replicating networks in an on-chip environment can improve performance while simultaneously improving area and energy efficiency. Topologies such as the Mesh and CMesh may accommodate a second network without increasing the die area, though the network will occupy more of the chip area. The placement of the routers and channels must be considered before estimating the area overhead of introducing a second network. Our results comparing different strategies for distributing traffic over the subnetworks illustrate the importance of equitably distributing load across the subnetworks.

Drawing on the above insights, we present the concentrated mesh architecture with replicated subnetworks and express channels. Concentration reduces hop counts and improves load balance without increasing wiring complexity. The regular structure efficiently exploits the abundant wiring resources available on chip by providing wide channels. Channel lengths are kept short to reduce energy dissipation. Express channels further improve energy efficiency by directing packets around routers while simultaneously improving performance by reducing hop counts between distant processors. Thoughtful planning allows us to introduce a second network without increasing die area. The result is a network that performs well while controlling area overhead and introducing a modest energy overhead.

Acknowledgments

We wish to acknowledge the contributions of Rebecca Schultz and thank the reviewers for insightful comments.

9. REFERENCES

[1] A. Adriahantenaina, H. Charlery, A. Greiner, L. Mortiez, and C. A. Zeferino. Spin: A scalable, packet switched, on-chip micro-network. In *DATE '03: Proceedings of the conference on Design, Automation and Test in Europe*, page 20070, Washington, DC, USA, 2003. IEEE Computer Society.

[2] P. Bai *et al.* A 65nm logic technology featuring 35nm gate lengths, enhanced channel strain, 8 cu interconnect layers, low-k ild and 0.57 μm^2 sram cell. In *Electronic Devices Meeting, 2004. IEDM Technical Digest*, pages 657–660. IEEE International, Dec 2004.

[3] A. Chatterjee *et al.* A 65 nm cmos technology for mobile and digital signal processing applications. In *Electronic Devices Meeting, 2004. IEDM Technical Digest*, pages 665–668. IEEE International, Dec 2004.

[4] W. J. Dally and B. Towles. Route packets, not wires: on-chip inteconnectoin networks. In *DAC '01: Proceedings of the 38th conference on Design automation*, pages 684–689, New York, NY, USA, 2001. ACM Press.

[5] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

[6] J. Duato. A new theory of deadlock-free adaptive routing in wormhole networks. *IEEE Trans. Parallel Distrib. Syst.*, 4(12):1320–1331, 1993.

[7] N. Eisley and L.-S. Peh. High-level power analysis for on-chip networks. In *CASES '04: Proceedings of the 2004 international conference on Compilers, architecture, and synthesis for embedded systems*, pages 104–115, New York, NY, USA, 2004. ACM Press.

[8] R. Ho, K. Mai, and M. Horowitz. The future of wires. In *Proceedings of the IEEE*, volume 89, pages 490–504. IEEE, April 2001.

[9] R. Ho, K. Mai, and M. Horowitz. Managing wire scaling: a circuit perspective. In *Proceedings of the IEEE 2003 International Interconnect Technology Conference*, pages 177–179, June 2003.

[10] International technology roadmap for semiconductors. 2005 edition.

[11] J. Kim, D. Park, T. Theocharides, N. Vijaykrishnan, and C. R. Das. A low latency router supporting adaptivity for on-chip interconnects. In *DAC '05: Proceedings of the 42nd annual conference on Design automation*, pages 559–564, New York, NY, USA, 2005. ACM Press.

[12] C. E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Trans. Comput.*, 34(10):892–901, 1985.

[13] Z. Luo *et al.* High performance and low power transistors integrated in 65nm bulk cmos technology. In *Electronic Devices Meeting, 2004. IEDM Technical Digest*, pages 661–664. IEEE International, Dec 2004.

[14] M. L. Mui, K. Banerjee, and A. Mehrotra. A global interconnect optimization scheme for nanometer scale vlsi with implications for latency, bandwidth, and power dissipation. In *IEEE Transactions on Electron Devices*, volume 51, pages 195–202. IEEE, February 2004.

[15] R. Mullins, A. West, and S. Moore. Low-latency virtual-channel routers for on-chip networks. In *ISCA '04: Proceedings of the 31st annual international symposium on Computer architecture*, page 188, Washington, DC, USA, 2004. IEEE Computer Society.

[16] S. R. Öhring, M. Ibel, S. K. Das, and M. J. Kumar. On generalized fat trees. In *IPPS '95: Proceedings of the 9th International Symposium on Parallel Processing*, page 37, Washington, DC, USA, 1995. IEEE Computer Society.

[17] K. Olukotun, B. A. Nayfeh, L. Hammond, K. Wilson, and K. Chang. The case for a single-chip multiprocessor. *SIGOPS Oper. Syst. Rev.*, 30(5):2–11, 1996.

[18] L.-S. Peh and W. J. Dally. A delay model and speculative architecture for pipelined routers. In *HPCA '01: Proceedings of the 7th International Symposium on High-Performance Computer Architecture*, page 255, Washington, DC, USA, 2001. IEEE Computer Society.

[19] D. Seo, A. Ali, W.-T. Lim, N. Rafique, and M. Thottethodi. Near-optimal worst-case throughput routing for two-dimensional mesh networks. *SIGARCH Comput. Archit. News*, 33(2):432–443, 2005.

[20] M. B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, and A. Agarwal. The raw microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE Micro*, 22(2):25–35, 2002.

[21] H. Wang, L.-S. Peh, and S. Malik. Power-driven design of router microarchitectures in on-chip networks. In *MICRO 36: Proceedings of the 36th annual IEEE/ACM International Symposium on Microarchitecture*, page 105, Washington, DC, USA, 2003. IEEE Computer Society.

[22] W. Zhao and Y. Cao. New generation of predictive technology model for sub-45nm design exploration. *ISQED*, 0:585–590, 2006.