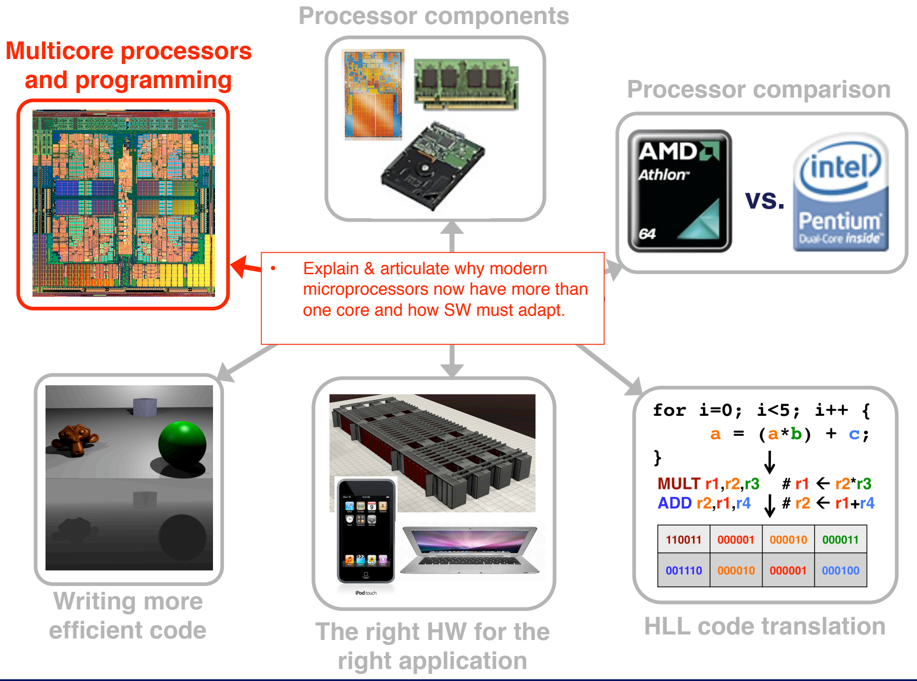
Lecture 28
On-Chip Interconnection Networks

Suggested Readings

- Readings
  - H&P: Chapter 7.8

Also…

- A few HW items
  - For Q2:
    - Determine % of problem that is parallelizable
    - Apply Amdahl's Law
  - For Q3:
    - Look at Lecture 26 slides on GPUs
  - For Q4:
    - Can assume MESI or MSI protocol
    - Can assume intervention or no intervention if requesting modified data
    - If just reading shared data, should assume it comes from memory


Processor components
Multicore processors and programming
Processor comparison
AMD Athlon 64
vs.
intel Pentium Dual-Core inside
Writing more efficient code
The right HW for the right application
HLL code translation

- Explain & articulate why modern microprocessors now have more than one core and how SW must adapt.

```
for i=0; i<5; i++ {
      a = (a*b) + c;
}
MULT r1,r2,r3    # r1 ← r2*r3
ADD r2,r1,r4     # r2 ← r1+r4
```

110011	000001	000010	000011
001110	000010	000001	000100

Impediments to Parallel Performance

- Reliability:
  - Want to achieve high "up time" – especially in non-CMPs
- Contention for access to shared resources
  - i.e. multiple accesses to limited # of memory banks may dominate system scalability
- Programming languages, environments, & methods:
  - Need simple semantics that can expose computational properties to be exploited by large-scale architectures
- Algorithms

Not all problems are parallelizable

$$\text{Speedup} = \frac{1}{\left[1 - \text{Fraction}_{\text{parallelizable}}\right] + \frac{\text{Fraction}_{\text{parallelizable}}}{N}}$$

What if you write good code for 4-core chip and then get an 8-core chip?

- Cache coherency
  - P1 writes, P2 can read
    - Protocols can enable $ coherency but add overhead

Overhead where no actual processing is done.

Challenges: Latency

- …is already a major source of performance degradation
  - Architecture charged with hiding local latency
    - (that's why we talked about registers & caches)
  - Hiding global latency is also task of programmer
    - (I.e. manual resource allocation)
- Today:
  - access to DRAM in 100s of CCs
  - round trip remote access in 1000s of CCs
  - multiple clock cycles to cross chip or to communicate from core-to-core
    - Not "free" as we assumed in send-receive example from L27

We'll talk more quantitatively about this today.

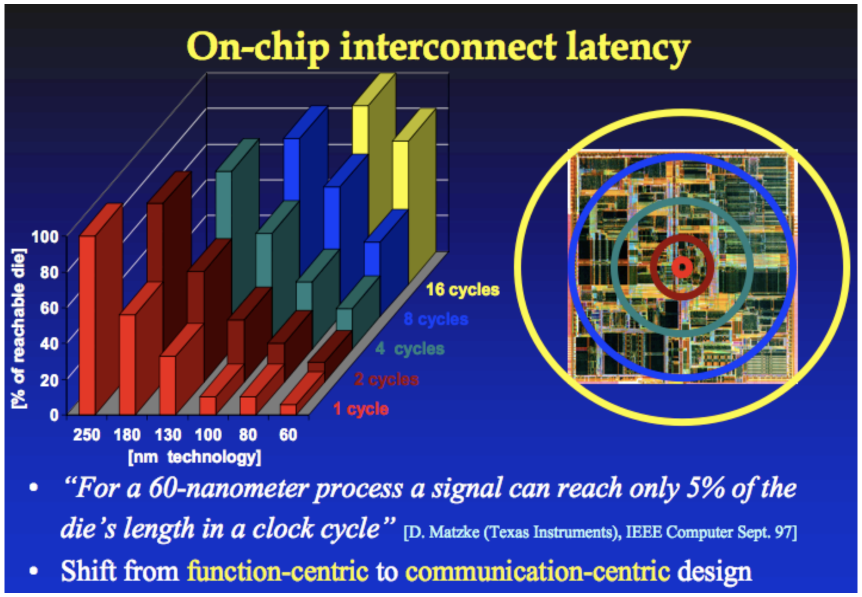
Overhead where no actual processing is done.

Impediments to Parallel Performance

- All ’ed items also affect $\text{Fraction}_{\text{parallelizable}}$
  - (and hence speedup)

$$\text{Speedup} = \frac{1}{\left[1 - \text{Fraction}_{\text{parallelizable}}\right] + \frac{\text{Fraction}_{\text{parallelizable}}}{N}}$$

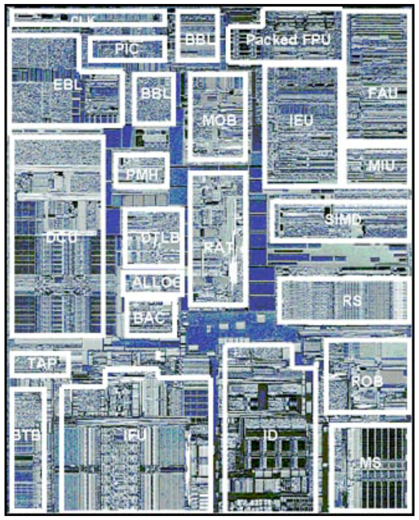
Some Perspective…


On-chip interconnect latency

- *"For a 60-nanometer process a signal can reach only 5% of the die's length in a clock cycle"* [D. Matzke (Texas Instruments), IEEE Computer Sept. 97]
- Shift from function-centric to communication-centric design

Pentium III Die Photo

Deterministic connections as needed.



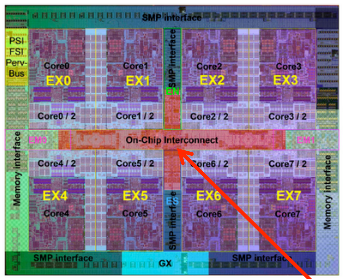
1st Pentium III, Katmai: 9.5 M transistors, 12.3 * 10.4 mm in 0.25-mi. with 5 layers of aluminum

- EBL/BBL - Bus logic, Front, Back
- MOB - Memory Order Buffer
- Packed FPU - MMX Fl. Pt. (SSE)
- IEU - Integer Execution Unit
- FAU - Fl. Pt. Arithmetic Unit
- MIU - Memory Interface Unit
- DCU - Data Cache Unit
- PMH - Page Miss Handler
- DTLB - Data TLB
- BAC - Branch Address Calculator
- RAT – Register Alias Table
- SIMD - Packed Fl. Pt.
- RS - Reservation Station
- BTB - Branch Target Buffer
- IFU - Instruction Fetch Unit (+I$)
- ID - Instruction Decode
- ROB - Reorder Buffer
- MS - Micro-instruction Sequencer

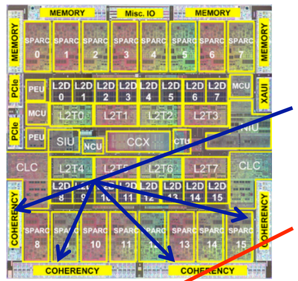
CPSC 614:Graduate Computer Architecture, TAMU, Prof. Lawrence Rauchwerger; Based on Lectures by Prof. David A. Patterson UC Berkeley

Recent multi-core die photos

(Route packets, not wires?)



http://dx.doi.org/10.1109/ISSCC.2010.5434074



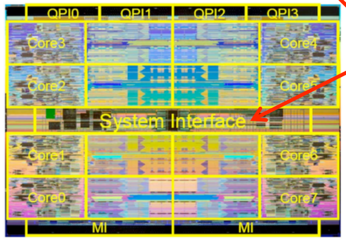
http://dx.doi.org/10.1109/ISSCC.2010.5434030

Likely to see HW support for parallel processor configurations:

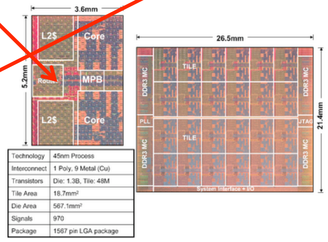
- Coherency

+

- On-chip IC NWs



http://dx.doi.org/10.1109/ASSCC.2009.5357230



Technology	45nm Process
Interconnect	1 Poly, 9 Metal (Cu)
Transistors	Die: 1.3B, Tile: 48M
Tile Area	18.7mm²
Die Area	567.1mm²
Signals	970
Package	1567 pin LGA package

http://dx.doi.org/10.1109/ISSCC.2010.5434077

…takes advantage of 8 voltage and 28 frequency islands to allow independent DVFS of cores and mesh. As performance scales, the processor dissipates between 25 W and 125 W. … 567 mm^2 processor on 45 nm CMOS integrates 48 IA-32 cores and 4 DDR3 channels in a 2D-mesh network. Cores communicate through message passing using 384 KB of on-die shared memory. Fine-grain power management

Take away from last 2 slides

- Cores communicate with each other
  - (and each others memory)
- Can no longer just realize direct, deterministic connection between processor’s functional units
- Fortunately, wide body of work to start with to enable more efficient/reasonable inter-core communication

Lot’s of history to leverage…

- Lot’s of XAN’s
  - SAN – system area network
    - Usually connects homogeneous nodes
    - Physical extent small – less than 25 m– usually much less
    - Connectivity usually from 100s to 1000s of nodes
    - Main focus is high bandwidth and low latency
  - LAN – local area network
    - Heterogeneous hosts assumed – designed for generality
    - Physical extent usually within a few hundred kms
    - Connectivity usually in the hundreds of nodes
    - Performance is typically mundane
    - Supported by workstation industry

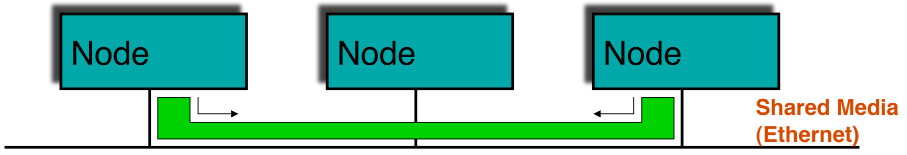
Lot's of history to leverage…

- WAN – wide area network
  - General connectivity for 1000s of heterogeneous nodes
  - High bandwidth but latency is usually horrible
  - Physical extent = K's of kilometers
  - Developed by the telecommunications industry

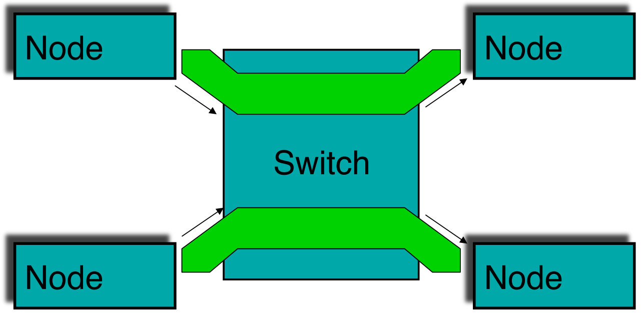
Can now think about doing all of the following "on-chip"

Creating shared media networks

- Messages are broadcast everywhere
  - Useful for cache coherency
  - Not unlike ethernet



Creating switched media networks



- Switches introduce switching overheads
- But, no time wasted on arbitration and collisions
- Multiple transfers can be in progress if different links used
- Circuit or Packet Switching
  - Circuit switching: end-to-end connections
    - Reserves links for a connection (e.g. phone network)
  - Packet switching: each packet routed separately
    - Links used only when data transferred (e.g. Internet Protocol)

Routing Messages

- Shared Media
  - Broadcast to everyone!
- Switched Media needs real routing. Options:
  - Source-based routing: *message* specifies path to the destination (changes of direction)
  - Virtual Circuit: circuit established from source to destination, message picks the circuit to follow
  - Destination-based routing: message specifies destination, switch must pick the path
    - deterministic: always follow same path
    - adaptive: pick different paths to avoid congestion, failures
    - randomized routing: pick between several good paths to balance network load

Routing Methods for Switches

- Store-and-Forward
  - Switch receives entire packet, then forwards it
- Wormhole routing
  - Packet consists of flits (a few bytes each)
  - First flit contains header w/ destination address
  - Switch gets header, decides where to forward
  - Other flits forwarded as they arrive
  - Looks like packet worming through network

Cut-Through Routing

- What happens when link busy?
  - Header arrives to switch, but outgoing link busy
  - What do we do with the other flits of the packet?
- Wormhole routing: stop the tail when head stops
  - Now each flit along the way blocks the a link
  - One busy link creates other busy links => traffic jam
- Cut-Through Routing
  - If outgoing link busy, receive and buffer incoming flits
  - The buffered flits stay there until link becomes free
  - When link free, the flits start worming out of the switch
  - Need packet-sized buffer space in each switch
    - Wormhole Routing switch needs to buffer only one flit

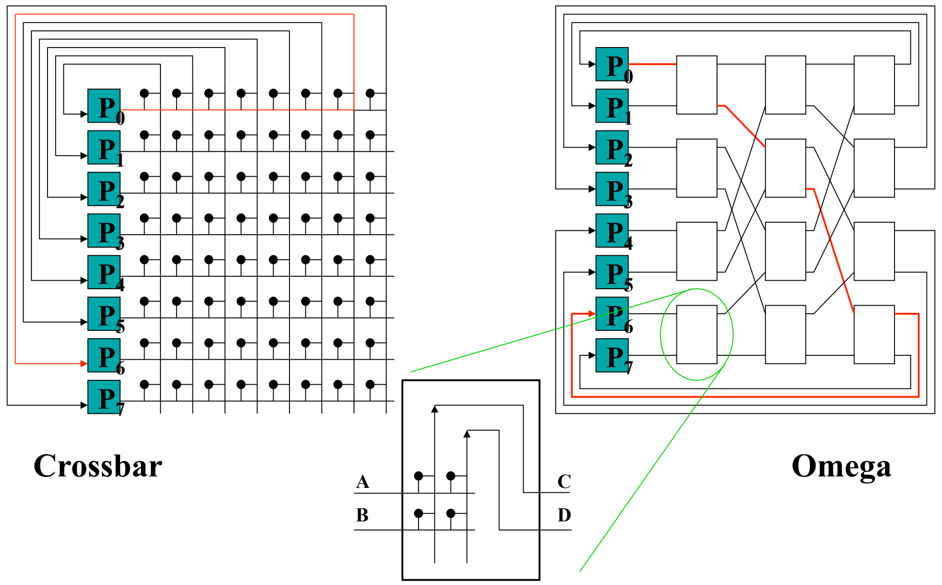
Wormhole vs. Cut Through

- In worm hole routing, when head of message is blocked, message stays strung out over the network, potentially blocking other messages (needs only buffer the piece of the packet that is sent between switches).
- Cut through routing lets the tail continue when head is blocked, accordioning the whole message into a single switch. (Requires a buffer large enough to hold the largest packet).

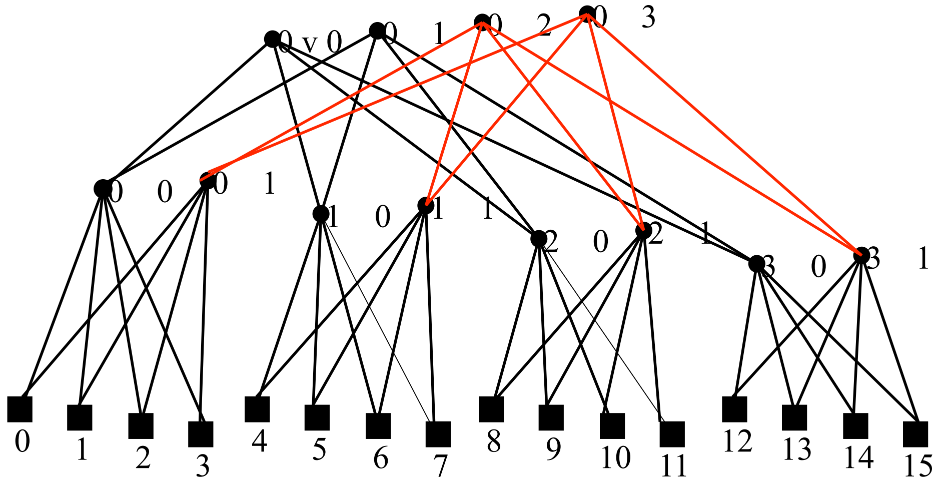
Switch Technology/Topologies

- Two common switching organizations
  - Crossbar
    - Allows any node to communicate with any other node with 1 pass through an interconnection
    - Very low switching delay, no internal contention
    - Complexity grows as square of number of links
      - Can not have too many links (i.e. 64 in, 64 out)
  - Omega
    - Uses less HW ($n/2 \log_2 n$ vs. n^2 switches); more contention
    - Build switches with more ports using small crossbars
    - Lower complexity per link, but longer delay and more contention

(crossbar vs. omega)



(others) Fat-tree topology

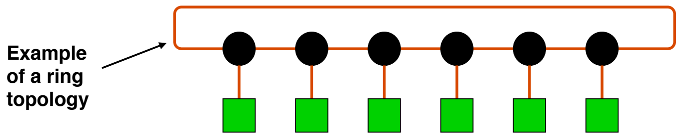


Circles = switches, squares = processor-memory nodes

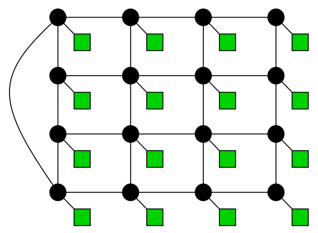
Higher bandwidth, higher in the tree – match common communication patterns

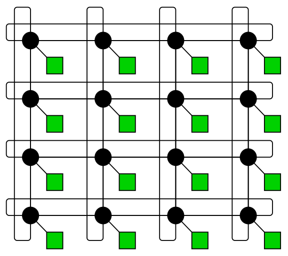
(others) Ring topology

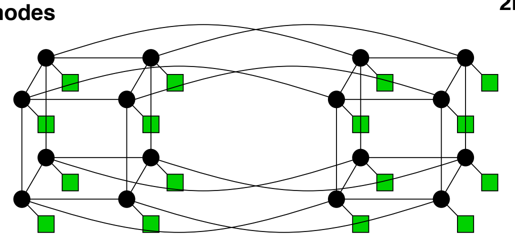
- Instead of centralizing small switching element, small switches are placed at each computer
  - Avoids a full interconnection network
- Disadvantages
  - Some nodes are not directly connected
    - Results in multiple "stops", more overhead
  - Average message must travel through $n/2$ switches, n = # nodes
- Advantages
  - Unlike shared lines, ring has several transfers going at once



(still others)


2D grid or mesh of 16 nodes

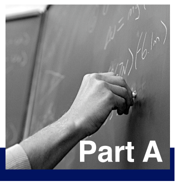

2D tour of 16 nodes


Hypercube tree of 16 nodes (16 = 2^4, so n = 4)

(others) Dedicated communication links

- Usually takes the form of a communication link between every *switch*
  - An expensive alternative to a ring…
  - Get *big* performance gains, but *big* costs as well
    - Usually cost scales by the square of the number of nodes
- The big costs led designers to invent “things in between”
  - In other words, topologies between the cost of rings and the performance of fully connected networks
- Whether or not a topology is “good” typically depends on the situation
- Some popular topologies for MPPs are
  - Grids, tours, hypercubes

Now, how might *real, on-chip* networks perform?



NW topologies

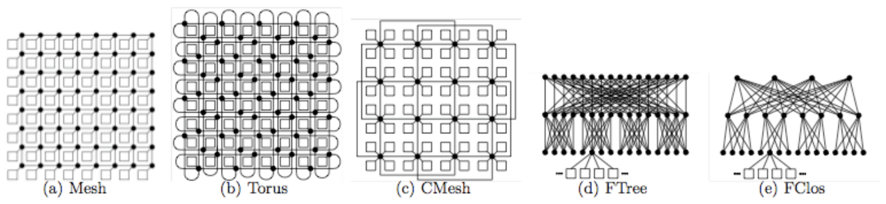


Figure 8: Network Topologies

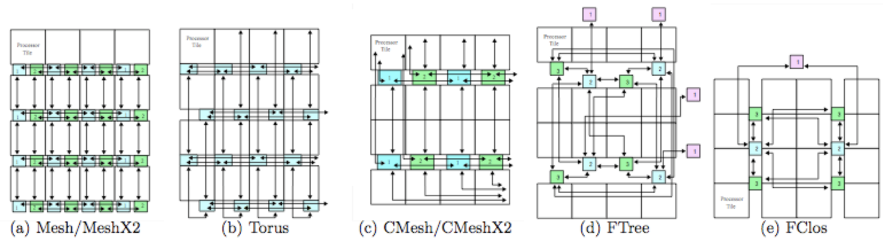
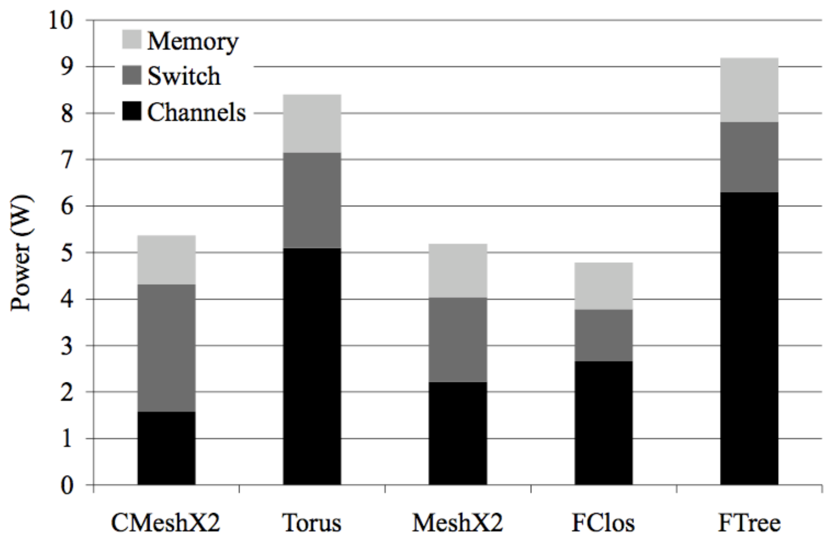
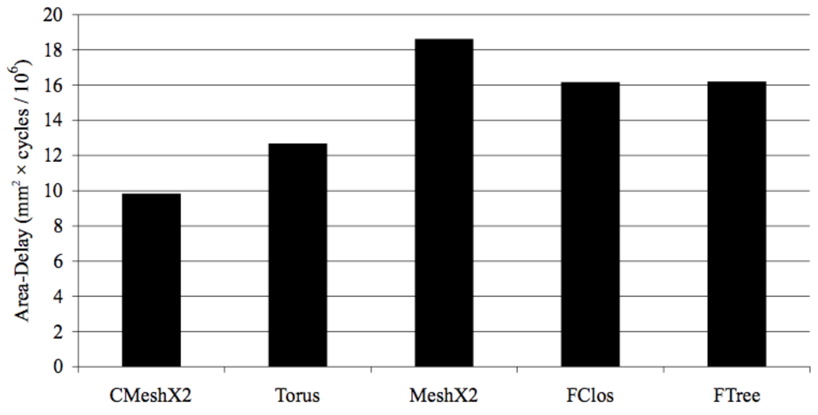


Figure 9: Placement of Routers used to Estimate Area (Lower Left Quadrant)

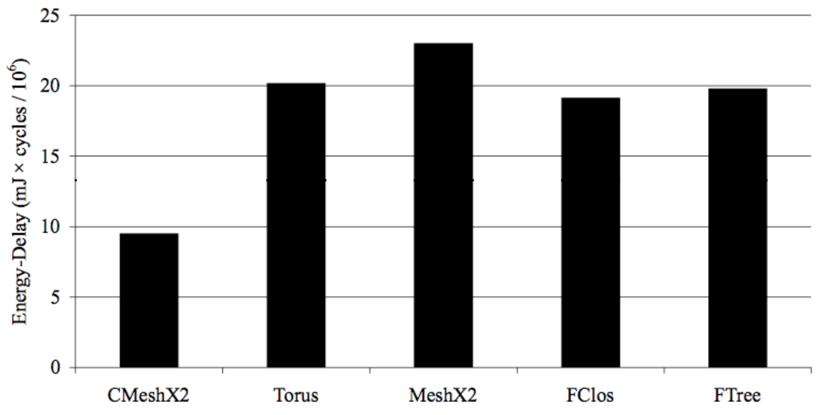
From Balfour, Dally, *Supercomputing*



(c) Network Power Dissipation



(d) Area Delay Metric



(e) Energy Delay Metric

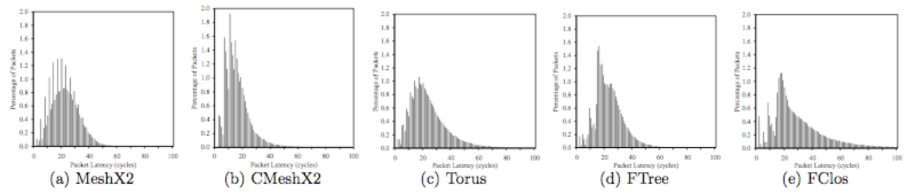


Figure 11: Workload Packet Latency Distribution for Uniform Random Traffic Pattern

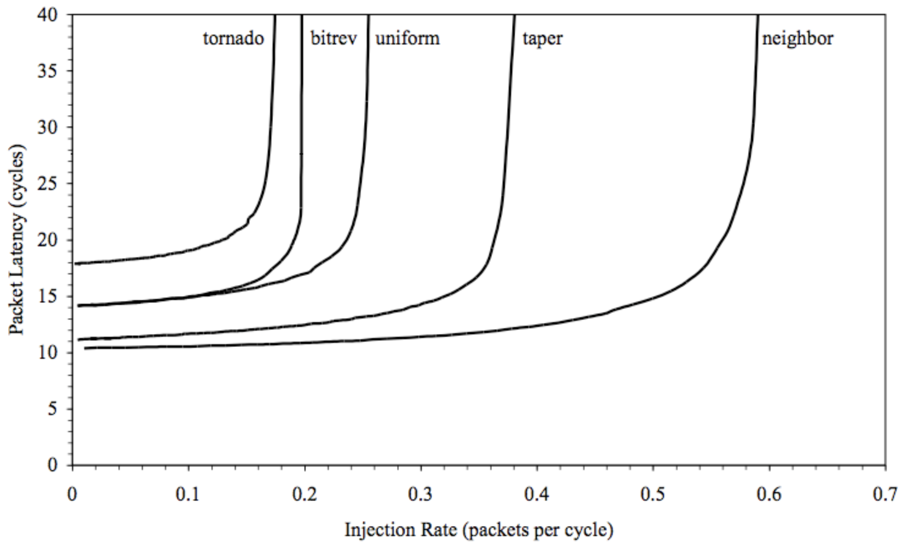


Figure 12: Offered Latency for CMeshX2 Network

From Balfour, Dally, *Supercomputing*