ESE680-002 (ESE534): Computer Organization

Day 22: April 4, 2007

Interconnect 7: Time Multiplexed Interconnect

Previously

- Multicontext computation
- Interconnect Topology
- Configured Interconnect
  - Lock down route between source and sink

Today

- Interconnect Style
  - Static Time Multiplexed
  - Dynamic Packet Switched
- Online/local vs. Offline/global

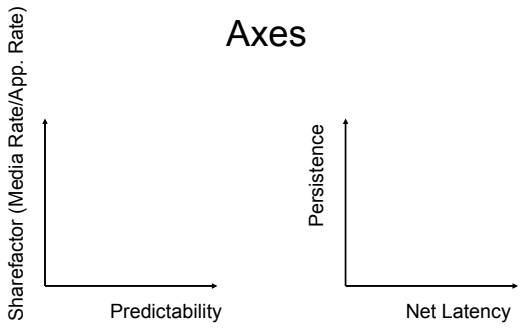
Motivation

- Holding a physical interconnect link for a logic path may be wasteful
  - Data Rate Logical Link < Physical Link
- *E.g.*
  - Time multiplexed logic
    - Logic only appears one cycle
  - Multirate
    - Some data may come at lower rates
  - Data dependent communication
    - Only need to send fraction of time

Issues/Axes

- Throughput of Communication relative to data rate of media
  - Single point-to-point link consume media BW?
  - Can share links between multiple comm streams?
  - What is the sharing factor?
- Binding time/Predictability of Interconnect
  - Pre-fab
  - Before communication then use for long time
  - Cycle-by-cycle
- Network latency vs. persistence of communication
  - Comm link persistence

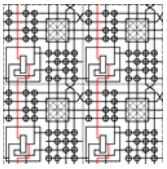
Axes

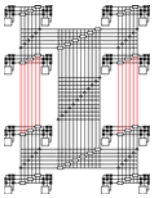


Hardwired

- Direct, fixed wire between two points
- *E.g.* Conventional gate-array, std. cell
- Efficient when:
  - know communication *a priori*
    - fixed or limited function systems
    - high load of fixed communication
      - often control in general-purpose systems
  - links carry high throughput traffic continually between fixed points

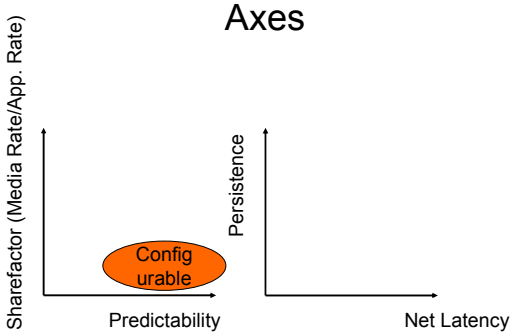
Configurable



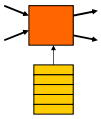


- Offline, lock down persistent route.
- *E.g.* FPGAs
- Efficient when:
  - link carries high throughput traffic
    - (loaded usefully near capacity)
  - traffic patterns change
    - on timescale >> data transmission

Axes

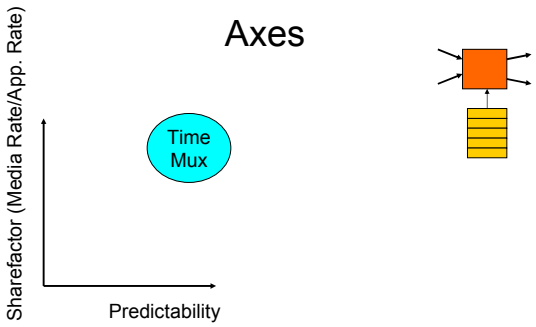


Time-Switched



- Statically scheduled, wire/switch sharing
- *E.g.* TDMA, NuMesh, TSFPGA
- Efficient when:
  - thruput per channel < thruput capacity of wires and switches
  - traffic patterns change
    - on timescale >> data transmission

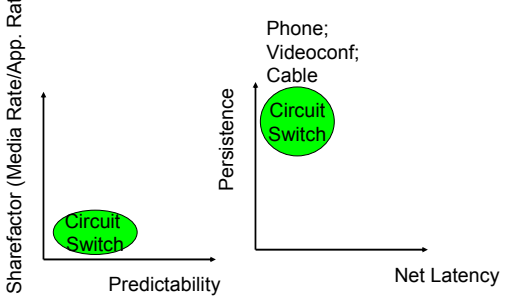
Axes



Self-Route, Circuit-Switched

- Dynamic arbitration/allocation, lock down routes
- *E.g.* METRO/RN1, old telephone net
- Efficient when:
  - instantaneous communication bandwidth is high (consume channel)
  - lifetime of comm. > delay through network
  - communication pattern unpredictable
  - rapid connection setup important

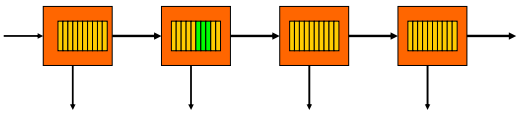
Axes



Self-Route, Store-and-Forward, Packet Switched

- Dynamic arbitration, packetized data
- Get entire packet before sending to next node
- *E.g.* nCube, early Internet routers
- Efficient when:
  - lifetime of comm < delay through net
  - communication pattern unpredictable
  - can provide buffer/consumption guarantees
  - packets small

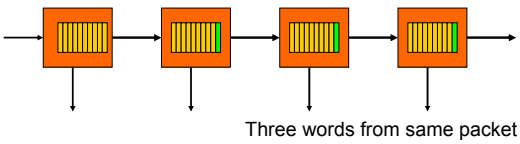
Store-and-Forward



Self-Route, Virtual Cut Through

- Dynamic arbitration, packetized data
- Start forwarding to next node as soon as have header
- Don't pay full latency of storing packet
- Keep space to buffer entire packet if necessary
- Efficient when:
  - lifetime of comm < delay through net
  - communication pattern unpredictable
  - can provide buffer/consumption guarantees
  - packets small

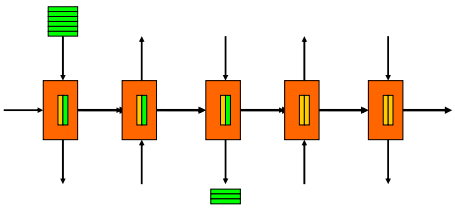
Virtual Cut Through



Self-Route, Wormhole Packet-Switched

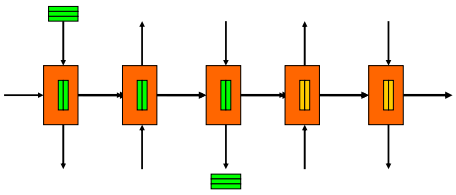
- Dynamic arbitration, packetized data
- *E.g.* Caltech MRC, Modern Internet Routers
- Efficient when:
  - lifetime of comm < delay through net
  - communication pattern unpredictable
  - can provide buffer/consumption guarantees
  - message > buffer length
    - allow variable (? Long) sized messages

Wormhole



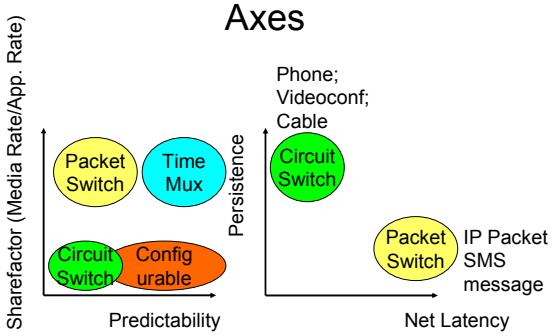
Single Packet spread through net when not stalled

Wormhole



Single Packet spread through net when stalled.

Axes



PS vs. TM

Following from
Kapre et al. / FCCM 2006

Time-Multiplexed (TM)

- Message paths are computed offline prior to execution
  - – Based on a workload specified already
- \+ Quality of route potentially better due to global view during routing
- – Need to store routing decisions in memory in hardware
- \+ Faster, simpler switches
- – Need to spend time computing routes offline
- – Need to known traffic pattern beforehand

Intuitive Tradeoff (TM)

- Benefit of Time-Multiplexing?
  - – Minimum end-to-end latency
  - – No added decision latency at runtime
  - – Offline route → high quality route
    - → use wires efficiently
- Cost of Time-Multiplexing?
  - – Route task must be static
    - Cannot exploit low activity
  - – Need memory bit per switch per time step
    - Lots of memory if need large number of time steps…

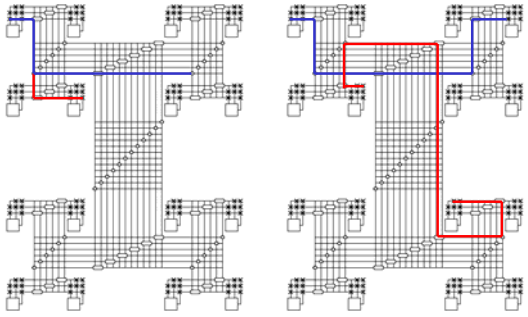
Packet-Switched (PS)

- Messages are dynamically routed on the network
  - Based on address in the packet
- – Complex switches (queues and address decode logic)
- – Known issues with deadlock/livelock/load-distribution (complex routing algorithms, poor route quality)
- \+ No prior knowledge of routes required,

Intuitive Tradeoff (PS)

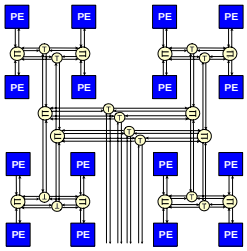
- Benefit of Packet Switching?
  - – No area proportional to time steps
  - – Route only active connections
  - – Avoids slow, off-line routing
- Cost of Packet Switching?
  - – Online decision making
    - Maybe won't use wires as well
    - Potentially slower routing?
      - – Slower clock **or** more clocks across net
  - – Data will be blocked in network
    - Adds latency
    - Requires packet queues (area)

Local Online vs. Global Offline



Butterfly Fat Trees (BFTs)

- Familiar from Day 15, 16
- Similar phenomena with other topologies
- Directional version



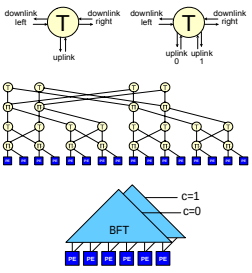
BFT Terminology

T = t-switch

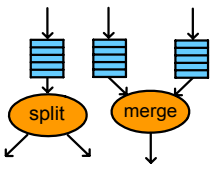
π = pi-switch

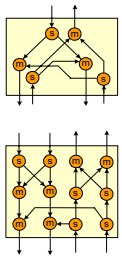
p = Rent Parameter
(defines sequence of T and π switches)

c = PE IO Ports
(parallel BFT planes)

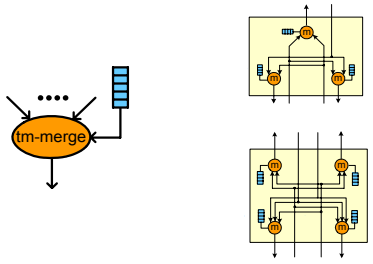


PS Hardware Primitives

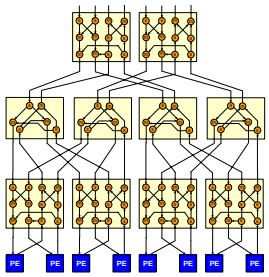




TM Hardware Primitives



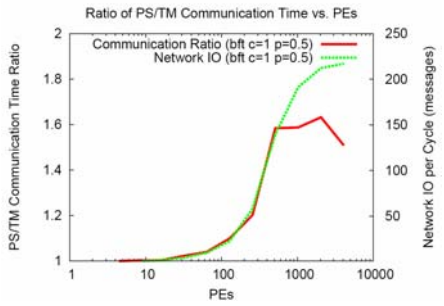
Complete Network



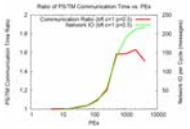
Analysis

- PS v/s TM for same topologies
  - Quantify inherent benefit of TM
- PS v/s TM for same area
  - Understand area tradeoffs (PEs v/s Interconnect)
- PS v/s TM for dynamic traffic
  - PS routes limited traffic, TM has to route all traffic

Iso-PEs



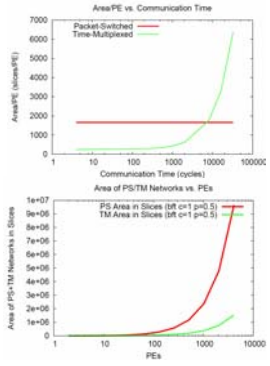
Iso-PEs



- PS vs. TM ratio at same PE counts
  - Small number of PEs little difference
    - Dominated by serialization (self-messages)
    - Not stressing the network
  - Larger PE counts
    - TM ~60% better
    - TM uses global congestion knowledge while scheduling

Area Effects

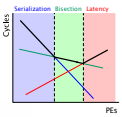
- Based on FPGA overlay model
- *i.e.* build PS or TM on top of FPGA

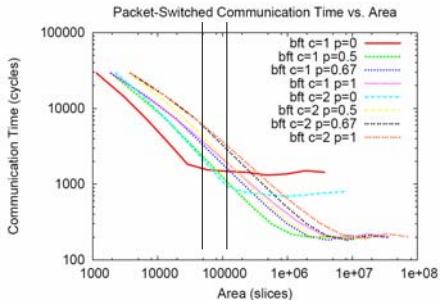


Area Analysis

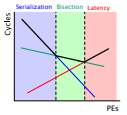
- Evaluate PS and TM for multiple BFTs
  - Tradeoff Logic Area for Interconnect
  - Fixed Area of 130K slices
    - p=0, BFT => 128 PS PEs => 1476 cycles
    - p=0.5, BFT => 64 PS PEs => 943 cycles
- Extract best topologies for PS and TM at each area point
  - BFT of different p best at different area points
- Compare performance achieved at these bests at each area point

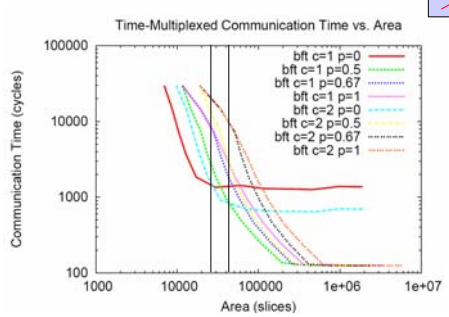
PS Iso Area: Topology Selection



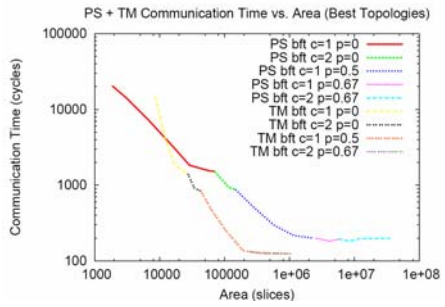


TM Iso Area

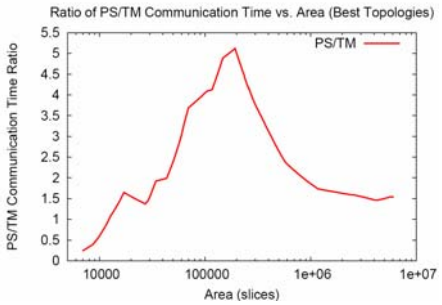




Iso Area

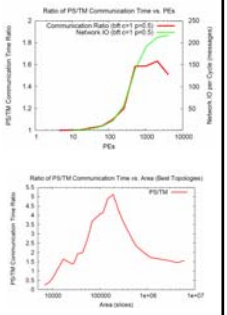


Iso Area



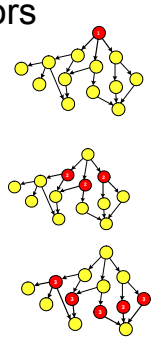
Iso Area

- Iso-PEs = TM 1~2x better
- With Area
  - PS 2x better at small areas
  - TM 4-5x better at large areas
  - PS catches up at the end
- Iso-Area = TM ~5x better

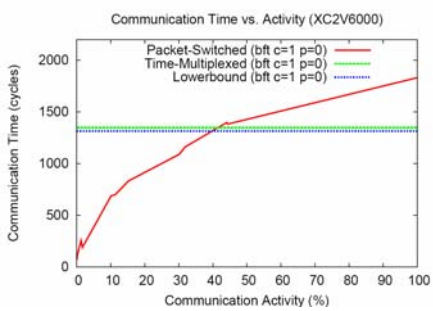


Activity Factors

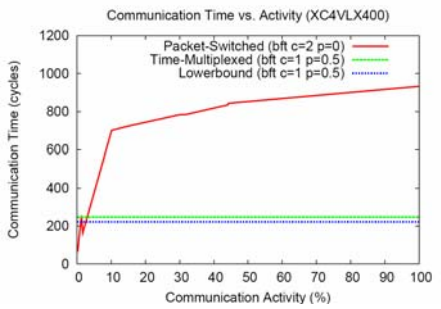
- Activity = Fraction of traffic to be routed
- TM needs to route all
- PS can route fraction
- Variable activity queries in ConceptNet
  - Simple queries ~1% edges
  - Complex queries ~40% edges



Activity Factors



Crossover could be less



Lessons

- Latency
  - PS could achieve same clock rate
  - But took more cycles
  - Didn't matter for this workload
- Quality of Route
  - PS could be 60% worse
- Area
  - PS larger, despite all the TM instrs
  - Big factor
  - May be "technology" dependent
    - Need to review for custom model
    - Will be smaller relative factor for custom

Admin

- Homework 8
- Final Exercise

Big Ideas [MSB Ideas]

- Different interconnect switching styles based on
  - Relative throughput, predictability, persistence, latency
- Low throughput → time share interconnect
- High predictability →
  - Efficiency for offline/global solutions
- Low predictability → dynamic