Slide 0

SPIDER: Flexible and Efficient Communication Support for Point-to-Point Distributed Systems

James Dolter, Stuart Daniel, Ashish Mehra,

Jennifer Rexford, Wu-chang Feng, and Kang Shin

June 24, 1994

Real-Time Computing Laboratory
The University of Michigan
Ann Arbor, Michigan

Slide 1

Outline

- Motivation
- Flexible routing and switching
- SPIDER architecture
- Protocol support
- Current status and conclusions

Slide 2

Real-Time Applications

Applications

- Industrial process control
- Computer-integrated manufacturing
- Fly-by-light/drive-by-wire

Communication requirements

- Bounded network access delay
- Resource allocation and scheduling
- Fault tolerance

Slide 3

Point-to-Point Distributed Systems

- Independent node and link failures
- Multiple disjoint routes between nodes
- High bandwidth
- Scalability

Electrical Engineering and Computer Science
The University of Michigan

Slide 4

Why Hardware Communication Support?

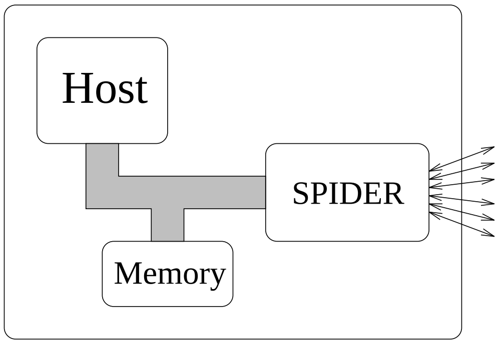
Low-level protocol processing burdens communication software:

- Multiple high-speed links
- Packet routing and switching
- Buffer management
- Data copying
- Error detection

Electrical Engineering and Computer Science
The University of Michigan

Slide 5

SPIDER (Scalable Point-to-point Interface DrivER) Node



Electrical Engineering and Computer Science
The University of Michigan

Slide 6

Why Flexible Communication Hardware?

- Irregular topologies
- Variable packet size
- Multiple communication protocols
- Diverse performance requirements
- Control over packet scheduling

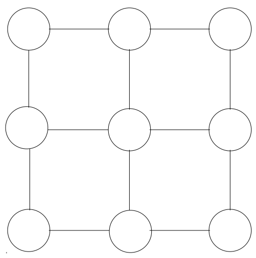
→ flexibility close to the physical links

Slide 7

Routing and Switching

Routing: Which nodes/links a packet traverses.

Switching: How the packet flows through nodes/links in its route.



→ tailor to application and system requirements

Slide 8

Flexible Routing

Flexible selection of output channels supports:

- Static routing
  → in-order delivery, predictable resource usage
- Adaptive routing
  → congestion-avoidance, fault tolerance
- Multicast
  → efficient group communication
- Irregular topologies
  → general networks, reconfiguration

Slide 9

Flexible Switching

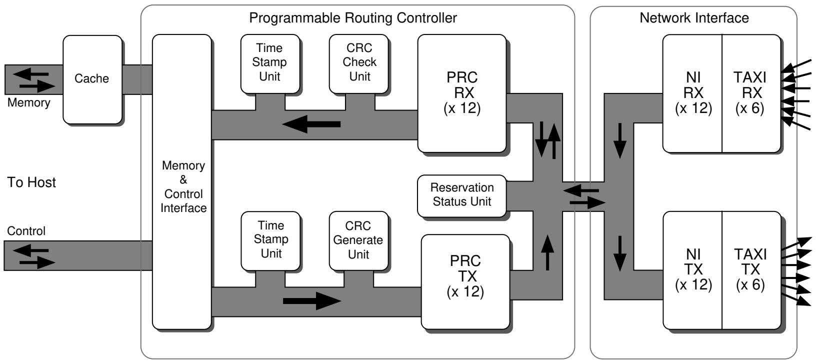
Flexible control over channel reservation facilitates:

- Circuit switching
  → guaranteed network access
- Packet switching
  → control over scheduling and bandwidth allocation
- Virtual cut-through switching
  → low latency, reduced intrusion from in-transit traffic
- Wormhole switching
  → low latency, complete insulation from in-transit traffic

Electrical Engineering and Computer Science
The University of Michigan

Slide 10

SPIDER Architecture



Electrical Engineering and Computer Science
The University of Michigan

Slide 11

Support for Efficient Protocol Processing

Host exercises *page*-level control over SPIDER:

- Page tag queues for each incoming/outgoing channel
- Event queue logging page-level activities

Flexibility:

- Scatter-gather DMA for each channel
- Variable packet size
- Partial cut-through
- Interrupt masking

Electrical Engineering and Computer Science
The University of Michigan

Slide 12

Support for Dependability

- Transparent CRC unit close to physical links
- Multicast primitive for group communication
- Multiple node-disjoint routes
- Fault-tolerant routing
- Irregular topologies

Electrical Engineering and Computer Science
The University of Michigan

Slide 13

Support for Real-Time Communication

- Timestamp unit close to physical links
- Fair, binary priority tree arbitration
- Fine-grain interleaving at memory and network interfaces
- Host scheduling for buffered packets
- Support for circuit switching
- Separate virtual networks

Electrical Engineering and Computer Science
The University of Michigan

Slide 14

Current Status and Future Work

- PRC chip designed and simulated using Epoch Verilog
- VME-based SPIDER board under development
- Cycle-level simulations of a mesh of SPIDERs
- Software protocols for point-to-point distributed systems

Slide 15

Conclusions

SPIDER insulates the software from low-level protocol processing without precluding control over network policies.

- Offloads processing from software protocols
- Exploits concurrency at the physical links
- Interfaces to a variety of topologies
- Implements multiple routing and switching schemes

Electrical Engineering and Computer Science
The University of Michigan