Introduction

- Linux Native APIC driver and User Mode Code Library Update
- Berkley Shands
- Washington University Applied Research Laboratory
Topics Today

- Native Linux Driver (TCP, AAL0, AAL5) Configuration
- Multiple APICs and PCI Bandwidth
- Default Pacer Clock to 70Mbytes/Sec
- User Mode Code Library Additional Demos

Native TCP/IP driver

- Uses ATM V0.78 signaling code
- Uses 10MB or 4MB (MEMHOG Parameter) for buffers of PageSize (4K or 8K)
- Uses a 32K default MTU for performance (nfs+)
- Atmarp --s <IP> 0.0.VCI for PVC
Multiple APICs and PCI Limits

- Linux Driver supports up to 4 APICs per machine
- An APIC/PCI Bus can send 98MBytes/Sec with NO other traffic
- Reading has Priority over Sending so an APIC can be output Starved in a high load situation
- APICs should be on different busses (PCI32 or PCI64)

Pacer Clock

- Pacer Clock Magic Number is “13” for about 70MBytes/Sec Throttle on Sending
- Derived from PacerClock.c results
- Reduces overload on switch bursts
- Can provide per VCI bandwidth pacing as a percent of Line Rate
User Mode Library

- Added PacerClock.c for generating pacer values
- Added AAL5Generator.c to create TCP/IP frames (pings) at a given rate and size
- Added ZeroCopy*.c for template code using Zero Copy Send
- Added low level routines

Conclusion

- Fully Functional Native Driver
- Concurrent User Mode Access
- NetBSD support for User Mode Code Library