

M-VIA and MVICH: Status and Future Plans

Michael Welcome

Paul Hargrove

Lawrence Berkeley National Lab

mlwelcome@lbl.gov

phhargrove@lbl.gov



Overview



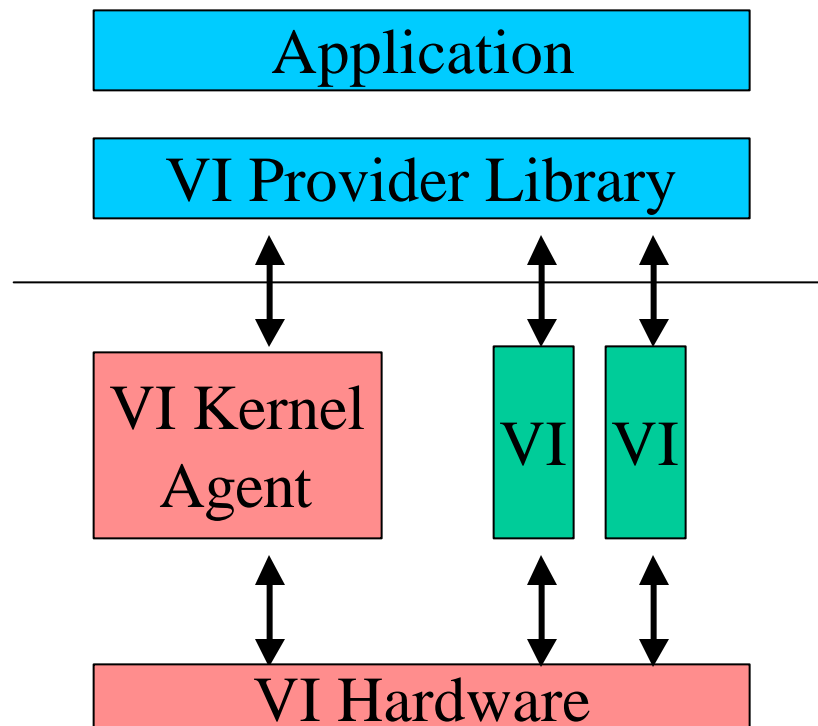
- VIA: Virtual Interface Architecture
- M-VIA: Modular VIA for Linux
- MVICH: Implementation of MPICH ADI-2 for VIA

Why VIA – Fast IPC for Clusters



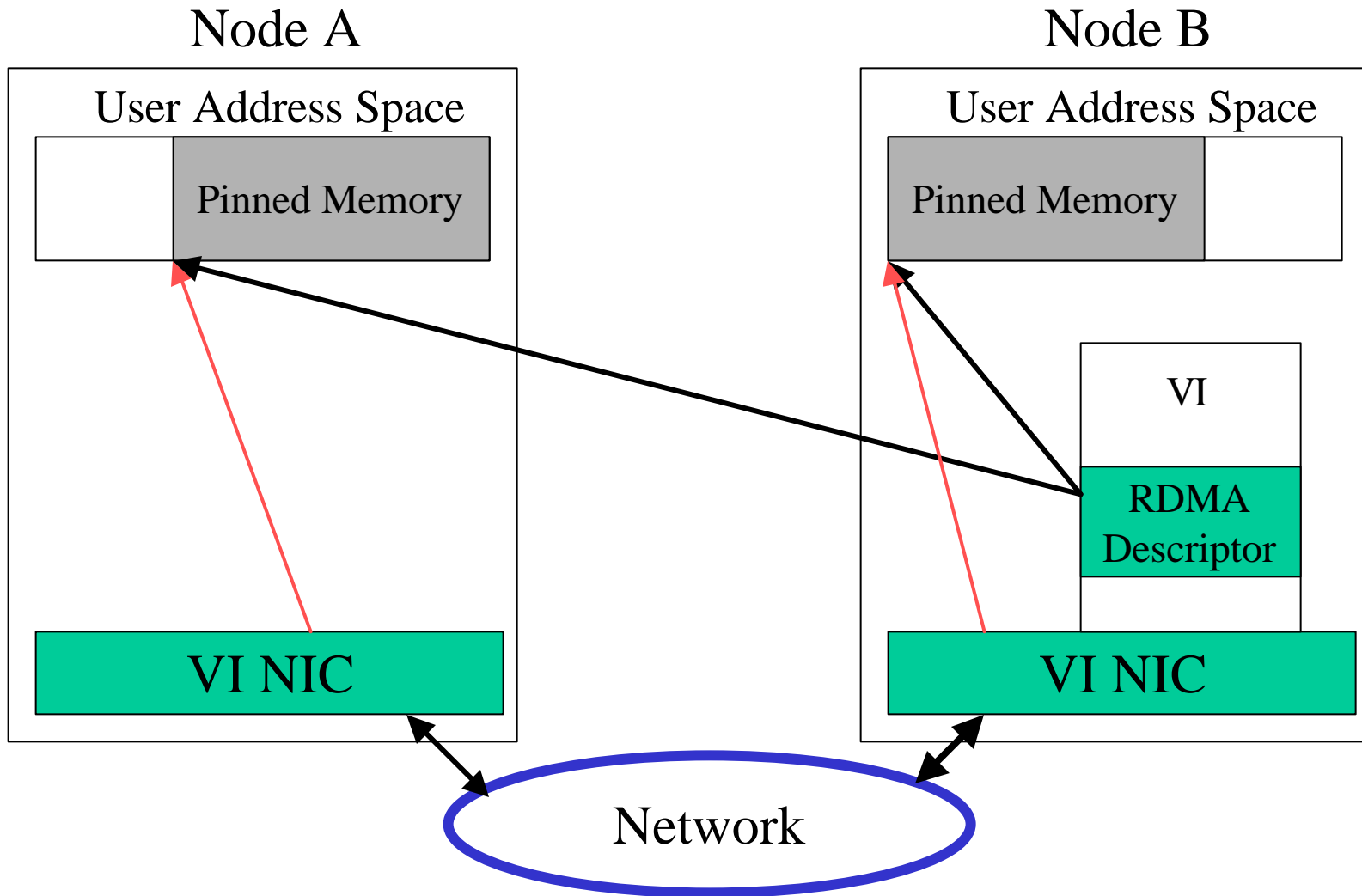
- Reduce message passing overhead incurred by traditional protocol stacks (TCP).
 - Thin software layer
 - OS Bypass
 - Asynchronous processing by intelligent NIC
- Industry Driven Standard – promoted by Intel, Compaq and Microsoft
 - VI Developers Forum (VIDF) – Stewardship of current standard.
 - API for application programmers (VIPL)
 - Defined Application-Hardware Interaction
- Commodity Networking Hardware
 - Giganet C-lan
 - Compaq Servernet II
 - Myricom – VIA over GM (alpha testing)

VI Architecture



- Standardized API (VIPL)
- VI – Communication Endpoint
 - Send and Recv Queue
 - Doorbell
 - Descriptor-mediated interaction with NIC
- VI NIC
 - Send/Recv or RDMA – No CPU interaction
 - Gather-Scatter of data segments
 - Requires “pinned” data regions
- VI Kernel Agent – Device Driver
 - Open/Close NIC
 - Connection Setup
 - Memory registration

VIA – RDMA Operation

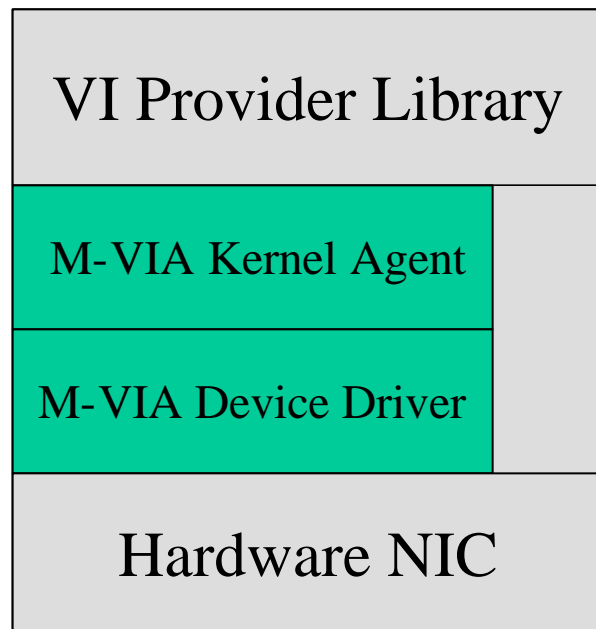


M-VIA: Modular VIA for Linux



- Goals:
 - Reference implementation
 - Emulated drivers for non-VIA aware NICs
 - Open Source – BSD style license
 - Compaq Servernet II
 - SysKonnnect
 - Portable to other Architectures
 - Thread safe
 - Kernel recompilation not required – Loadable Modules
 - Co-exist with traditional network protocols
 - Promote rapid development of new drivers

M-VIA Architecture



- VIPL implementation is driver independent – IOCTL and Fast Trap
- Kernel Agent is further abstracted:
 - Device Independent Kernel Agent
 - Connection Manager
 - Protection Tag Manager
 - Registered Memory Manager
 - Error Queue Manager
 - Device Dependent Drivers (Emulated)
 - Registers with Kernel Agent
 - May replace kernel agent managers
 - Device Classes allow similar drivers to share code

M-VIA: Status



- M-VIA 1.0 Released September 1999
- Implements all VIA functionality except peer-to-peer
- Passes Intel conformance test suite (100K lines of code)
- Emulated drivers supported
 - Loop-back – Fast on-node process to process copy
 - Fast Ethernet:
 - Tulip chipset
 - Intel EE Pro/100
 - Gigabit Ethernet:
 - Packet Engines G-NIC I and II
- Main developer quit 12/99 – replaced 7/00
- 9/00 Group leader left for DOT.COM

M-VIA 1.1 and 1.X



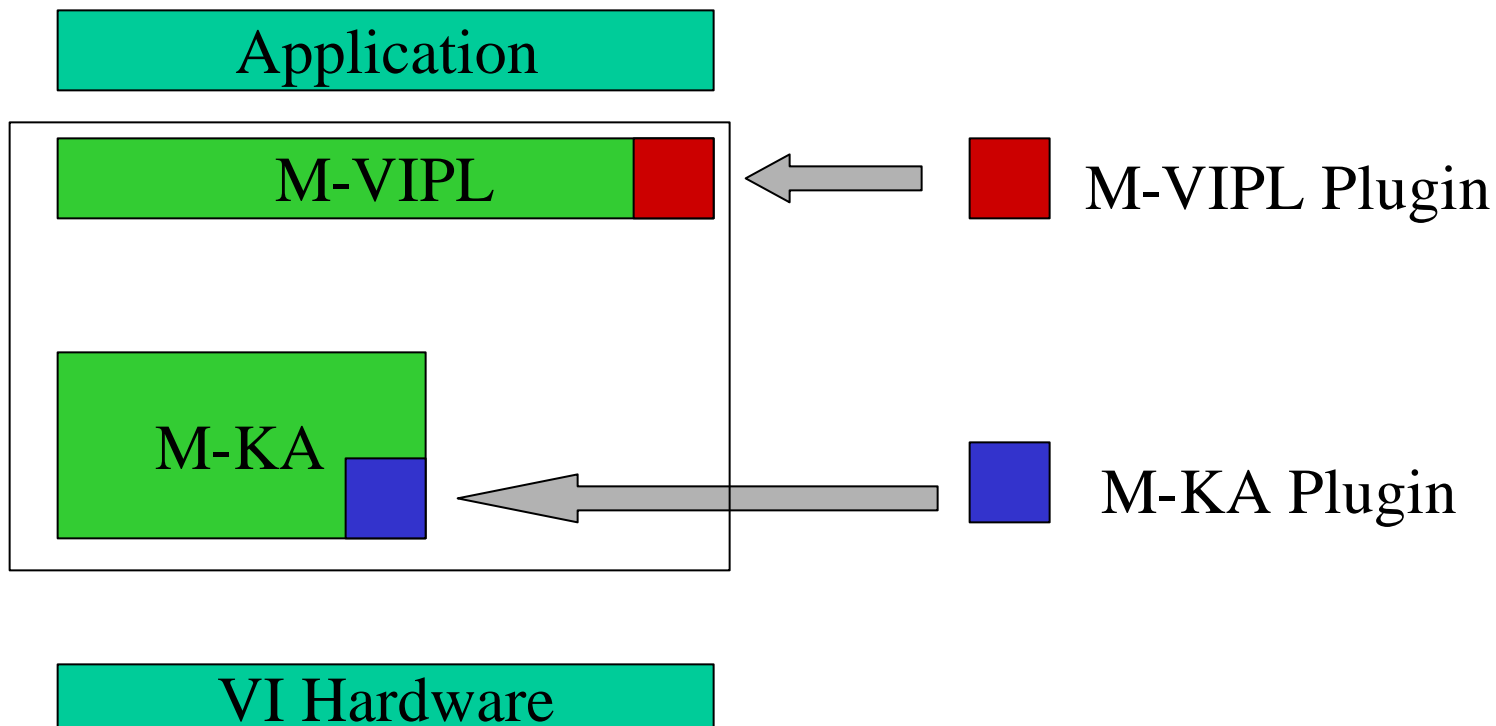
- M-VIA 1.1 – Before 1/01
 - Bug Fixes – Memory Leak
 - Peer-to-peer connections implemented (VIA option).
 - More Drivers
 - 3-COM Fast Ethernet
 - Intel Gigabit Ethernet
- M-VIA 1.X – (date unknown)
 - Linux 2.4 Kernel support
 - VIPL 1.1 support (when available)
 - Reliable Delivery
 - IA64 support (when available)

M-VIA 2



- The real target for M-VIA is VIA-aware networks
 - M-VIA 1 design based on pre-release 0.9X VIA spec
 - VIA 1.0 Specification relaxed specified interaction between VIPL and NIC
- M-VIA 2 has better support for VIA NICs
- Introduces modularity at user level.
- Improved internal interfaces based on phase 1 feedback

M-VIA 2 Plugins

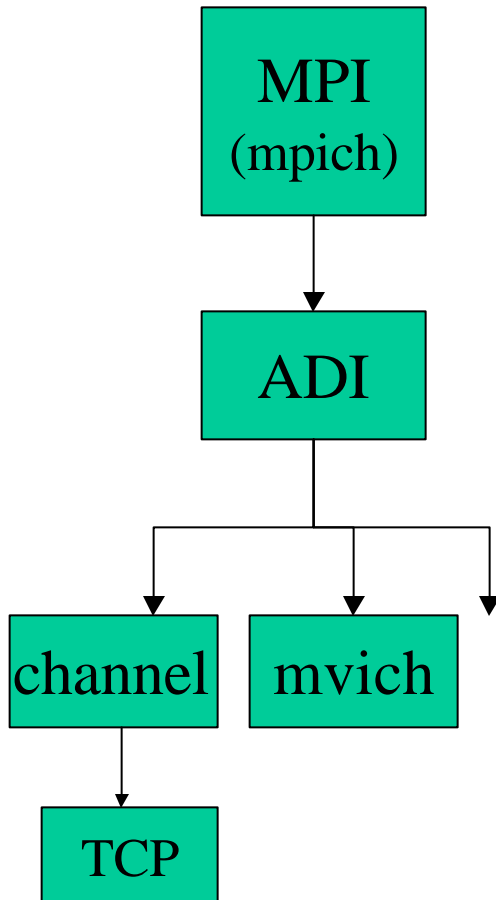


M-VIA 2 Features



- Plugins are dynamically loaded
 - User-level and kernel level loading
 - Multiple simultaneous plugins possible
- Applications do not need to be recompiled or relinked to use different NICs, *even though applications interact directly with the NIC*
 - Installation easier
 - Software distribution much easier
 - Example: Single application could use multiple devices:
 - M-VIA loop-back for communication within node
 - hardware NIC(s) for communication with other nodes.
- Status: in design stage, some code written.

MVICH



“devices”

- MVICH is an implementation of the MPICH ADI-2 for VIA
- Implements point-to-point message passing
- From-scratch implementation of ADI-2
 - No channels, no chameleon
 - Multi-device support stripped out
- To Build:
 - Put MVICH source tree in MPICH/mpid/via
 - Configure with your VIA device
 - Build

MVICH Implementation



- N-1 VI's created on each node, one for each node-to-node communication channel.
- Buffering
 - “vbufs” are VIA memory-registered MPI-managed buffers
 - Contain control info and, in certain cases, message data
- Flow control – VIA recv must be posted before send.
 - Credit scheme implements accounting system for pre-posted recvs.
 - Initially, each node pre-posts M recv vbufs on each VI, senders given M credits on each VI.
 - Sender decrements credit on send.
 - Receiver posts another vbuf after recv, “refresh” credits are piggybacked on rendezvous acknowledgements.
 - Credit scheme throttles sender

Protocols



- Short/Eager
 - Send data in one or more packets through vbufs
 - Requires buffering on receiver
- “R3” Rendezvous
 - Standard 3-way rendezvous through vbufs with pipelining
- “Rput” rendezvous
 - Zero copy RDMA write from sender to receiver
- “Rget” rendezvous
 - Zero copy RDMA read by receiver
- Rput and Rget revert to R3 if either side fails to register the user’s data area.

Dynamic memory registration



- Memory registration (pinning) is relatively expensive.
- In many MPI applications, same buffers are reused, even if persistent send/recv not used
- Basic scheme
 - Register on first use. Cache info.
 - Quickly ($\ll 1\mu\text{s}$) detect registration on subsequent use
 - Fall back to rendezvous through vbufs if necessary
 - Unregister after disuse (LRU)
 - MUST insure virtual-to-physical mapping has not changed between uses. Mvich uses mallopt options.

Process manager interface



- Drawback of MPICH/ch_p4 and LAM: communication and process management are closely tied.
- A number of projects are developing low-overhead process startup/management systems.
- Need to be able to plug together different process managers with communication systems.
- MVICH has a simple interface that lets it interact with any reasonable process manager.

Performance



- Current version is untuned and has known inefficiencies.
- Loopback: VIA latency 3.9us; MVICH latency 9us
- Giganet: VIA latency 8us; MVICH latency 14.5 us
- Giganet: VIA BW 90 MB/s; MVICH BW 71 MB/s
Note: this is the “slow” protocol with two copies.

Status



- Alpha release 1.0a6.1 release Sept 2000.
- Implemented
 - All basic infrastructure, including dynamic registration,
 - Eager, R3, Rput and Rget
 - Runs on M-VIA, Giganet VIA and Servernet II M-VIA
- Passes all but 5 MPICH conformance tests
 - Many bugs fixed since 1.0a5 release
 - MPI_CANCEL for SEND is not implemented (3 failures)
 - Aborts, rather than returning control to error handler in some cases. (2 failures – truncation tests)
- Correctly runs NAS benchmarks.

MVICH – Future Plans



- 1.0 release targeted for Jan 2001
 - Add tuning hooks – settable at run time
 - Add statistics package to trace behavior
 - Run “real” application, eg. adaptive mesh CFD.
- MVICH-2 (no estimated date)
 - Multi-threaded implementation
 - Support for unreliable networks
 - Asynchronous communication

Production/Commercial MPI



- MPI Software Technologies makes MPI for VIA on NT and Linux (www.mpi-softtech.com)
- Some advantages:
 - Support
 - Base implementation targeted for VIA (MPICH is “general”).
 - Thread safe
 - Full asynchronous communication
 - Datatype optimizations

For more info...



- <http://www.nersc.gov/research/ftg/via>
- via@nersc.gov (LBL developers)
- mvich-users@george.lbl.gov