Overview of New Features for CN5000

Douglas Fuller, VP Software Engineering
MUG 2024

CORNELIS™
NETWORKS

About Cornelis Networks

Provider of open, intelligent, high-performance networking solutions designed to accelerate the world's most demanding AI and HPC applications

Omni-Path Software

- Switch firmware and software
- Fabric Manager (opa-fm)
  - Subnet manager
  - Performance manager
  - Fabric executive
- Linux kernel driver (hfi1)
  - In-tree
  - Supports OPA100 and CN5000 HFIs
- Omni-Path Express (OPX)
  - libfabric provider
- User tools (Fast Fabric)
- SST / libfabric integration

Omni-Path + MVAPICH

- Nanoseconds before it was cool
  - Stampede + Omni-Path + MVAPICH < 1µs
- Forward to libfabric/OPX
  - Even lower (!) latency
- First-class citizen in testing
- CN5000 + MVAPICH
  - Early deliveries for Stampede
  - Early access for MVAPICH testing/optimization

Kernel API Changes

- SR-IOV implementation
- Driver remains the same (hfi1; supports both generations of devices)
- Some features require simple API changes
  - Generally not visible to middleware (`writev()` interface remains)
- Maintain uverbs for now, may change this in the future
- Driver communication needs a file descriptor (DMA engines, etc.)
  - Might switch to uverbs for this
- `io_uring()`
  - We've been experimenting
  - Slow adoption from distros
  - Will move when we're confident
- OPX libfabric provider already has these changes
- LKML RFCs in flight

libfabric

- A software project of the OpenFabrics Interfaces Working Group (ofiwg)
- Communication API for high-performance networking
- Maps application semantics to underlying hardware features
- Providers define fabric services in a structured way
  - One/two-sided messages, counters, barriers, built-in collectives, etc.
- API users (like MVAPICH!) then consume these services with zero overhead
- Open source, community developed, widely used
- Our newest provider (`prov/opx`) will support both devices

Omni-Path Express (OPX)

- Custom libfabric provider for CN5000
  - Available for OPA100 (…and runs great under MVAPICH!)
- Supports a broad range of libfabric primitives
  - Please reach out if there are additions you need
- GPU support via `fi_hmem`
  - AMD and NVIDIA acceleration for free
  - …or do it yourself like MVAPICH
- Updating for libfabric 2.0
- Interoperates between OPA100 and CN5000

OPX Low-level tracing capability

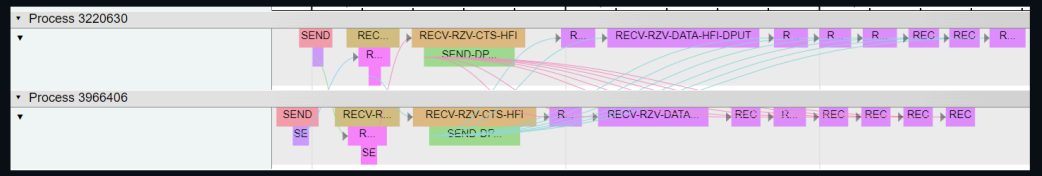
- Similar to mpiP but for lower-level libfabric primitives
- Tracepoints defined for most interesting libfabric operations
  - Open source, so you can add your own!
  - We're happy to help/extend as needed
- Use cases include performance profiling on new platforms/accelerators

OPX Tracer

Lindsay Reiser edited this page last week · 4 revisions

Introduction/Description

OPX Tracer is a performance profiling and visualization tool built into the OPX provider. It allows OPX developers to view the latency and sequence of events across multiple processes in benchmarks. For example, here is a single iteration of `IMB Sendrecv` using rendezvous (2 processes, 1 process per node, 64kB messages):



In the image, instrumented functions in OPX appear as blocks on the timeline called "events". The arrows, called "flow events", connect each event and mimic the data travelling across the wire. In this example, we can see the RZV protocol play out on each process. They initiate the communication by sending an RTS, respond with a CTS, and then send/receive the data packets (8x8kB packets=64kB). Pretty nifty!

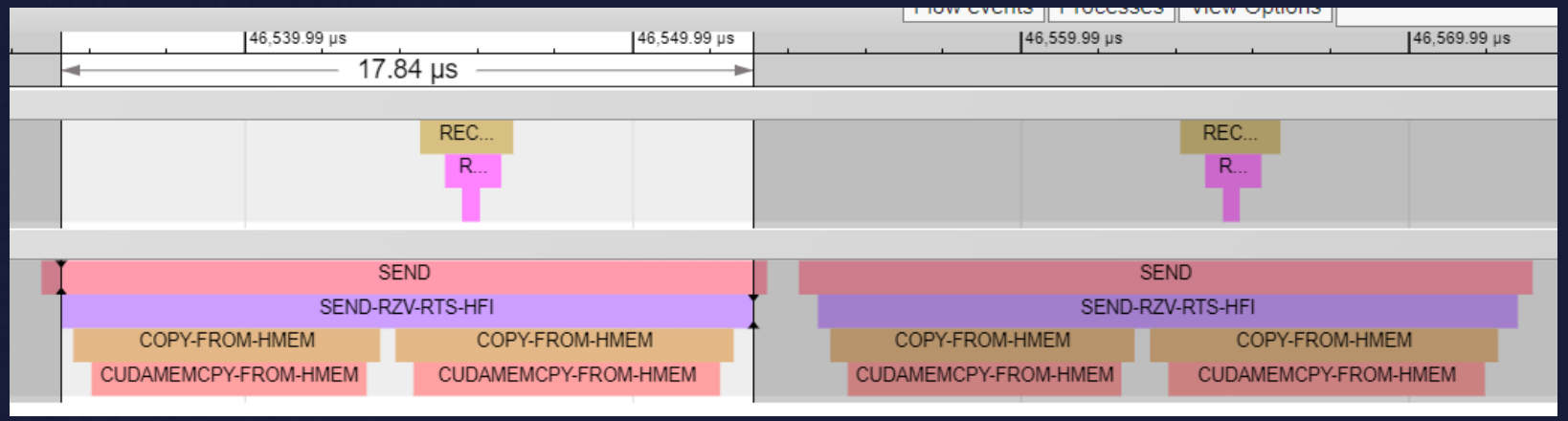
How to Build/Use

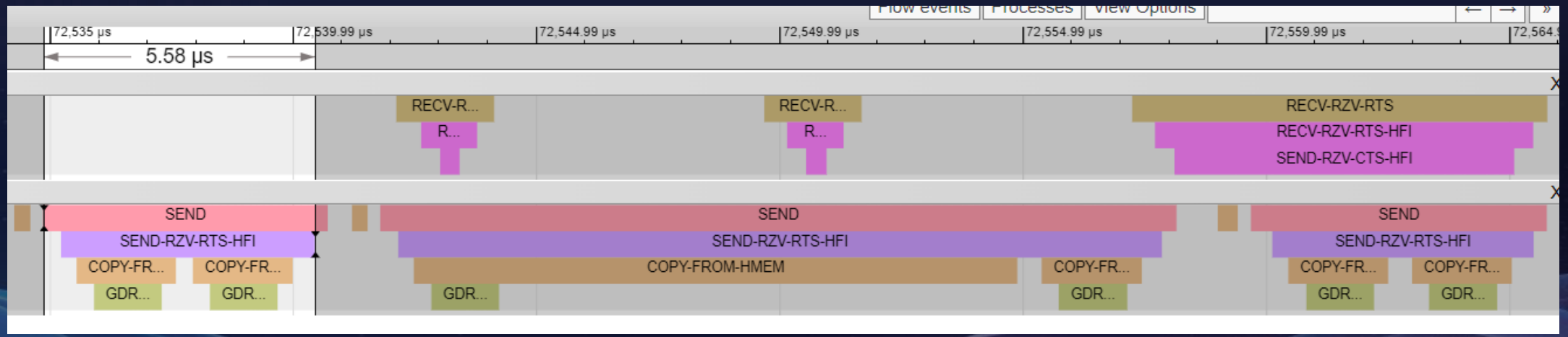
There are **two steps** to enable OPX Tracer:

1. Compile OPX w/ `-DOPX_TRACER` added to `CPPFLAGS`. If you are using the build script from `libfabric-devel`, just add `-d OPX_TRACER` to your build command.
2. Set the `FI_OPX_TRACER_OUT_PATH` environment variable to the path where you want to store the log files (e.g., `FI_OPX_TRACER_OUT_PATH=~/opx_tracer_logs`). I typically do this by appending it to the front of my `mpiexec` command.

After following these steps, run the command you want to profile, and a separate log file for each process will be written to `FI_OPX_TRACER_OUT_PATH`.

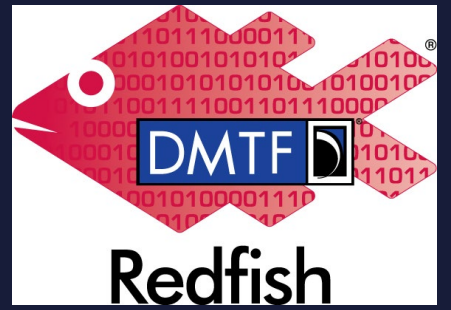
OPX Low-level tracing capability - example

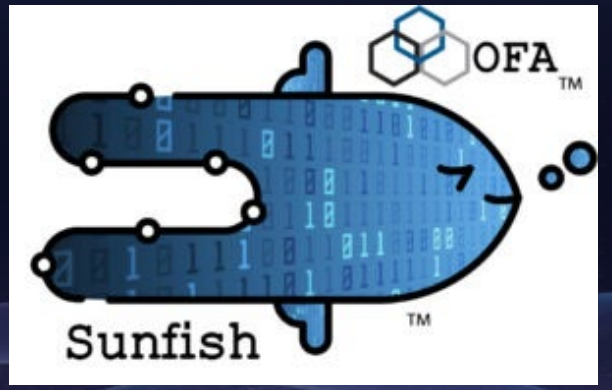




Network Health and Monitoring

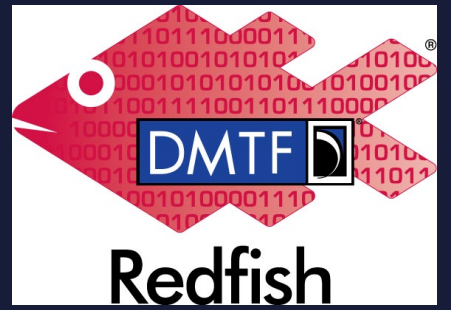
- New switch health and management API
  - Based on DTMF Redfish and OFA Sunfish
  - Already some software that plugs into this
- Congestion telemetry
- Lots of data available programmatically
  - We'll still provide the usual tools
  - Collaborating with the community to cross-pollinate more ideas
- OpenBMC-based switch firmware allows for customization
- Collaborating with the community for direct LDMS support
  - OPA100 Collectors will still work
  - Enhancements planned

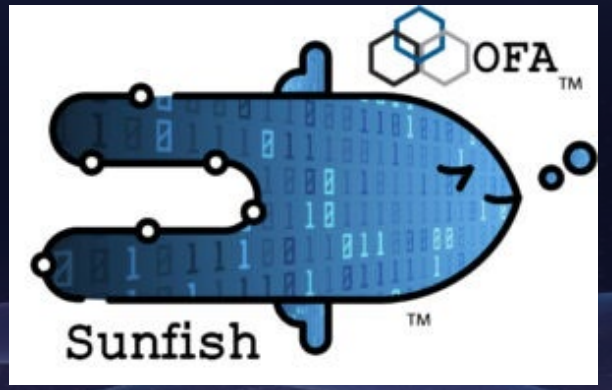




Network Health and Monitoring

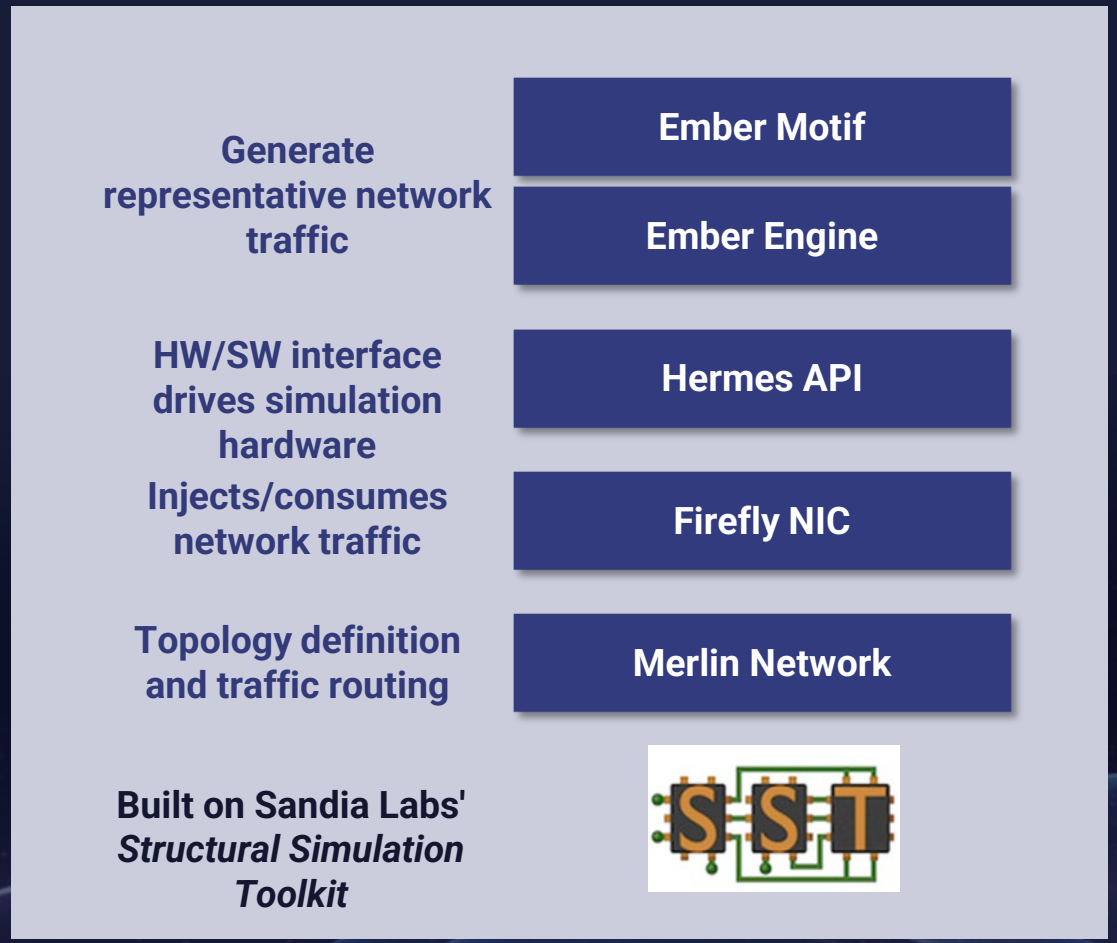
- Performance Manager (PM) and Performance Management Agent (PMA)
  - FM components that continuously monitor fabric health and performance – no manual sweeping
  - Tools for users (opareport) to perform direct queries (API coming soon)
    - `opareport -o errors -o slowlinks`
    - `opareport -o topology`
    - `opareport -o nodes -F lid:1`
    - `opareport -o route -S node:'node1 hfi1_0' -D node:'node2 hfi1_0'`
    - `opatop`
- What else would be interesting to middleware developers?





Network Simulation – Structural Simulation Toolkit

- Enables low-level simulation with models for network hardware, CPU, and memory
  - Think of it a bit like a Multiphysics code for network simulation
- Developed at Sandia National Laboratories
- Open source with a thriving community
- Contributions:
  - Additional models of interest to the community
  - Specific model for Cornelis hardware
- Collaboration welcome



New features for messaging libraries (or applications!)

User-controlled routing	**Header field determines minimal, dispersive, or fine-grained adaptive routing**
Receive-side matching rules	**Configurable, hardware-implemented packet inspection** **Delivers matching packets to a particular receive context**
Receive cut-through	**Land packets with zero buffering**
Receive bypass	**Allocate large portions of contiguous host memory to land continuous flows**

CORNELIS™
NETWORKS

Let’s Talk!

dfuller@cornelisnetworks.com