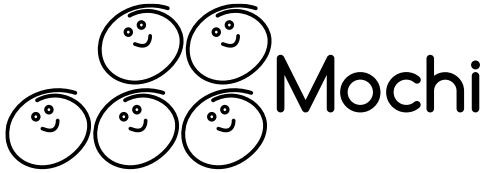
Mercury Updates

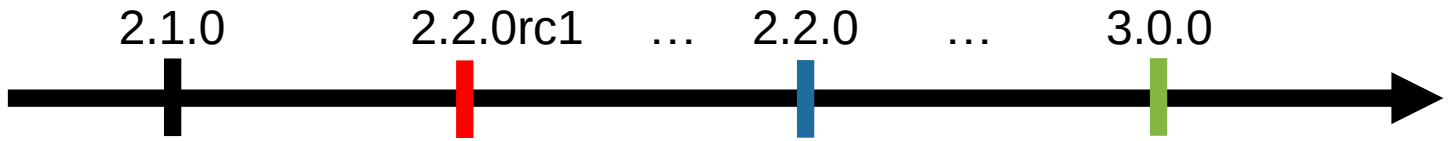
Mercury



- Base low-level RPC component used for communication between Mochi services
  - Always consider higher-level components first before directly using the mercury API
- In-depth documentation:
  - https://mercury-hpc.github.io

- Two main data transfer methods
  - Point-to-point RPC through eager messages
    - Connection-less semantics
  - Bulk data through RDMA
    - No memory copy
    - Requires memory registration internally

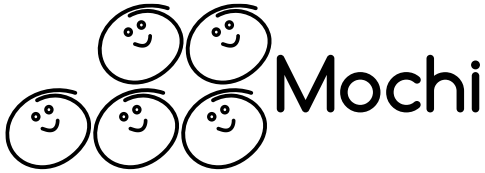
Status and Roadmap



Mochi

- 2.1.0 version released
  - Added initial support for UCX
  - Bug fixes
- 2.2.0rc1 version released
  - OFI / UCX:
    - Better handling of addressing formats and support for IPv6
    - Support device (CUDA, ROCm) to host transfers
  - OFI:
    - Support HPE Slingshot 11 through cxi provider
    - Support NIC locality through hwloc
  - UCX:
    - Switch to active messages for RPC requests
  - PSM/PSM2 (new plugin to support OmniPath)
    - Hopeful to have psm2 supported through OFI opx provider in the future
  - Improved diagnostics through `diag` log subsystem and improved OFI provider selection information
  - Checksums disabled by default
    - Introduced checksum levels
- 3.0.0 version
  - Extend addressing capabilities to address contexts (enhanced multithreading support and composability)
    - Improved support to OFI scalable endpoints

Supported Transports

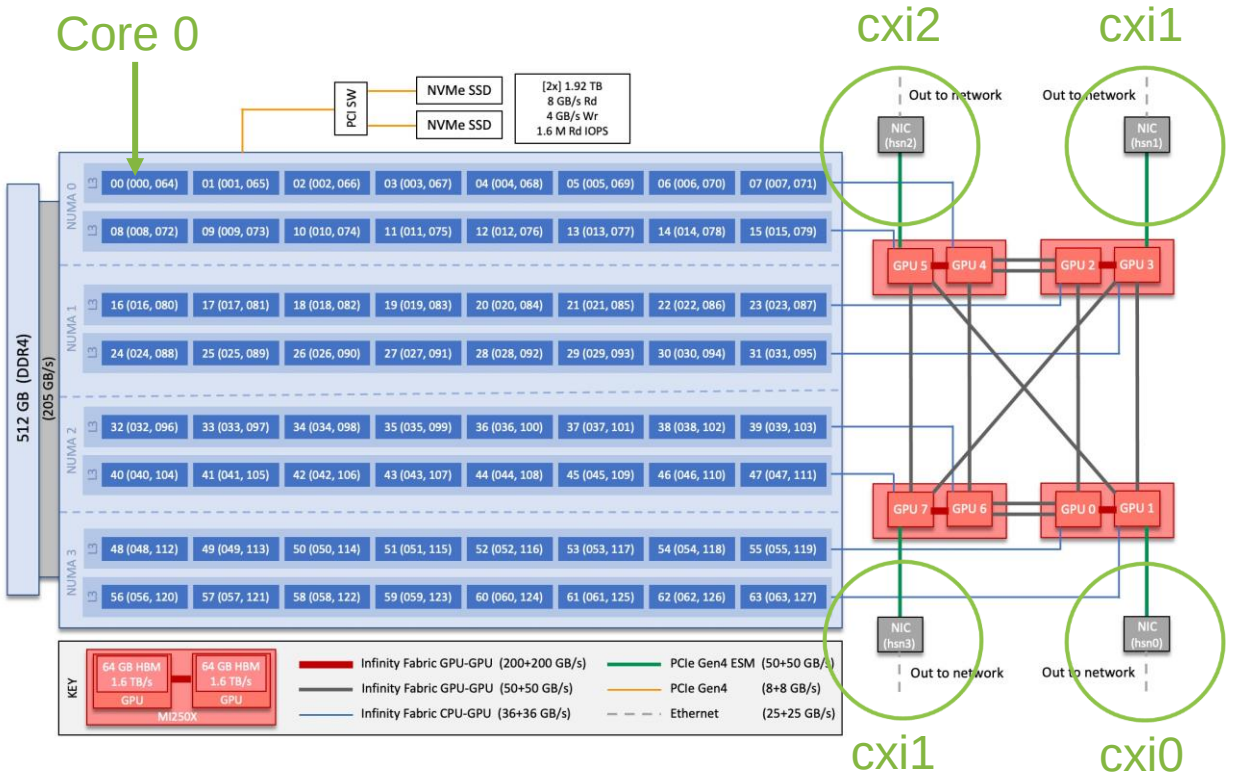


	tcp	verbs	shm	psm	psm2	gni	cxi
OFI	✓	✓	✗*	✗*	✓	✓	✓
UCX	✓	✓	✗*	✗	✗	✗*	✗
SM	✗	✗	✓	✗	✗	✗	✗
PSM	✗	✗	✗	✓	✓	✗	✗
BMI	✓	✗	✗	✗	✗	✗	✗

* Not explicitly supported by mercury but may be supported by underlying library

Slingshot Support and Locality Awareness

- Slingshot 11 supported w/ OFI cxi provider
  - Support only native addressing (i.e., no IP)
    - "ofi+cxi://cxi[0-9]:[0-510]"
  - All Mercury features supported by cxi provider except blocking progress
    - Busy spinning progress at the moment but will be resolved in a future libfabric update
- Locality awareness
  - Enabled when no interface is explicitly selected
    - "ofi+cxi://:[0-510]" or "ofi+cxi"
  - Uses PCI NIC information from libfabric and hwloc output to match closest NIC
- As for Cray GNI, communication between separate jobs may require key exchange (still under evaluation)

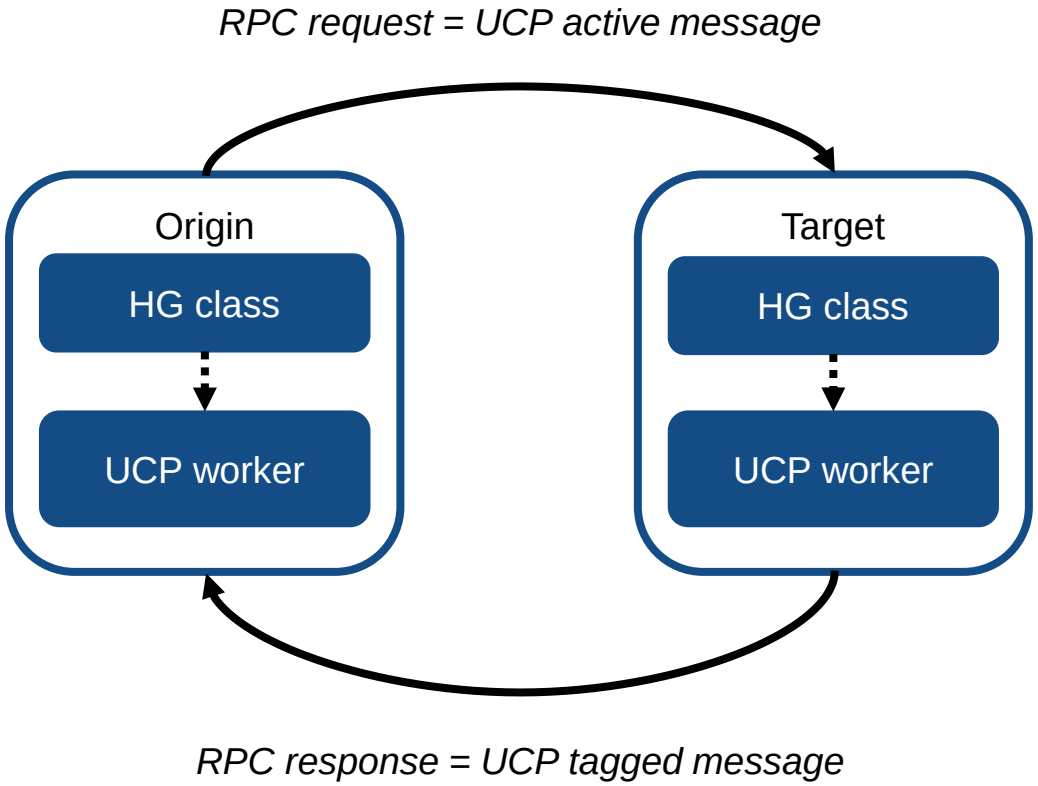


Credit: https://docs.olcf.ornl.gov/systems/crusher_quick_start_guide.html#system-overview

UCX

Mochi

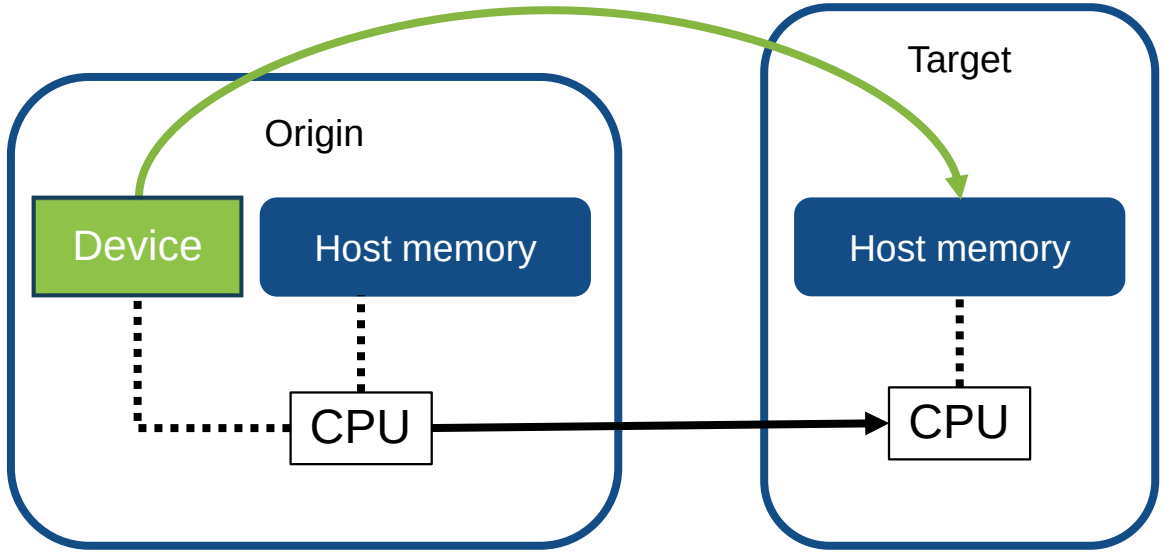
- Relies on UCP API of UCX
  - Combines both active and tagged messages
  - Supports native RDMA for bulk data
- All features of Mercury now supported
  - Only tested using tcp and verbs (in general ~1us faster than OFI on verbs)
- Supports only IP type of addressing
  - ucx+all://<hostname, IP, iface>:port
    - Recommended to always use "all" and let UCX decide on best protocol to use
- Thread safety mode can be relaxed w/ init info
  - Default is thread-safe
- Additional options passed through UCX environment variables



Host to Device RDMA Transfers

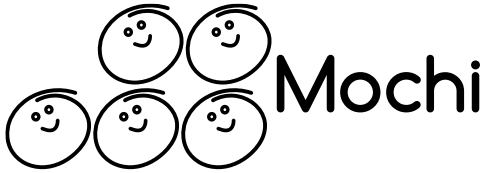
- New routine to provide memory type information on bulk handle creation
  - HG_Bulk_create_attr() with HG_MEM_TYPE_CUDA, HG_MEM_TYPE_ROCM, etc (default is HG_MEM_TYPE_HOST)
  - Supported by both OFI and UCX plugins
    - Only verbs and cxi for OFI
    - Transparent for UCX
- RPC requests and response are always sent between CPUs
  - Eager bulk transfers disabled when using device memory to prevent extra copy from device to CPU
- More testing remains to be done

Bulk transfer from origin device to target host memory



RPC request
(always sent from CPU to CPU)

Logging and Diagnostics




```
export HG_LOG_LEVEL=debug
export HG_LOG_SUBSYS=diag
```

```
### ---------------------
### (diag) counter log summary
### ---------------------
# Counters
# rpc_req_sent_count: 60 [RPC requests sent]
# rpc_req_recv_count: 0 [RPC requests received]
# rpc_resp_sent_count: 0 [RPC responses sent]
# rpc_resp_recv_count: 58 [RPC responses received]
# rpc_req_extra_count: 0 [RPCs with extra bulk request]
# rpc_resp_extra_count: 1 [RPCs with extra bulk response]
# bulk_count: 1 [Bulk transfers (inc. extra bulks)]
# -
```

Diagnostics counters can tell you about the type of RPCs that were sent / received

Debug output for OFI info give information about OFI provider
Compare with fi_info output

```
export HG_LOG_LEVEL=debug
export HG_LOG_SUBSYS=cls
```

(Similar debug output for UCX)

```
## na_ofi_verify_info(): FI info for selected provider
---
fi_info:
    caps: [ FI_RMA, FI_TAGGED, FI_READ, FI_WRITE, FI_RECV, FI_SEND, FI_REMOTE_RE
AD, FI_REMOTE_WRITE, FI_MULTI_RECV, FI_LOCAL_COMM, FI_REMOTE_COMM, FI_SOURCE, FI
_DIRECTED_RECV ]
    mode: [  ]
    addr_format: FI_SOCKADDR_IN
    src_addrlen: 16
    dest_addrlen: 0
    src_addr: fi_sockaddr_in://192.168.122.1:0
    dest_addr: (null)
    handle: (nil)
    fi_tx_attr:
        caps: [ FI_RMA, FI_TAGGED, FI_READ, FI_WRITE, FI_SEND ]
        mode: [  ]
        op_flags: [ FI_COMPLETION, FI_INJECT_COMPLETE ]
        msg_order: [ FI_ORDER_RAR, FI_ORDER_RAW, FI_ORDER_RAS, FI_ORDER_WAW, FI_
ORDER_WAS, FI_ORDER_SAW, FI_ORDER_SAS, FI_ORDER_RMA_RAR, FI_ORDER_RMA_RAW, FI_OR
DER_RMA_WAW, FI_ORDER_ATOMIC_RAR, FI_ORDER_ATOMIC_RAW, FI_ORDER_ATOMIC_WAW ]
        comp_order: [ FI_ORDER_NONE ]
        inject_size: 16384
        size: 65536
        iov_limit: 4
        rma_iov_limit: 4
    fi_rx_attr:
        caps: [ FI_RMA, FI_TAGGED, FI_RECV, FI_REMOTE_READ, FI_REMOTE_WRITE, FI_
MULTI_RECV, FI_SOURCE, FI_DIRECTED_RECV ]
        mode: [  ]
        op_flags: [ FI_COMPLETION ]
        msg_order: [ FI_ORDER_RAR, FI_ORDER_RAW, FI_ORDER_RAS, FI_ORDER_WAW, FI_
ORDER_WAS, FI_ORDER_SAW, FI_ORDER_SAS, FI_ORDER_RMA_RAR, FI_ORDER_RMA_RAW, FI_OR
DER_RMA_WAW, FI_ORDER_ATOMIC_RAR, FI_ORDER_ATOMIC_RAW, FI_ORDER_ATOMIC_WAW ]
        comp_order: [ FI_ORDER_NONE ]
        total_buffered_recv: 0
        size: 65536
        iov_limit: 4
    fi_ep_attr:
        type: FI_EP_RDM
        protocol: FI_PROTO_RXM
        protocol_version: 1
        max_msg_size: 18446744073709551615
        msg_prefix_size: 0
```