# **Bare Metal Library**

#### Abstractions for modern hardware Cyprien Noel







#### Plan

- 1. Devices, lots of them
- 2. It's a problem
- 3. But solutions
  - Device-centric abstractions
  - Device-to-device flows





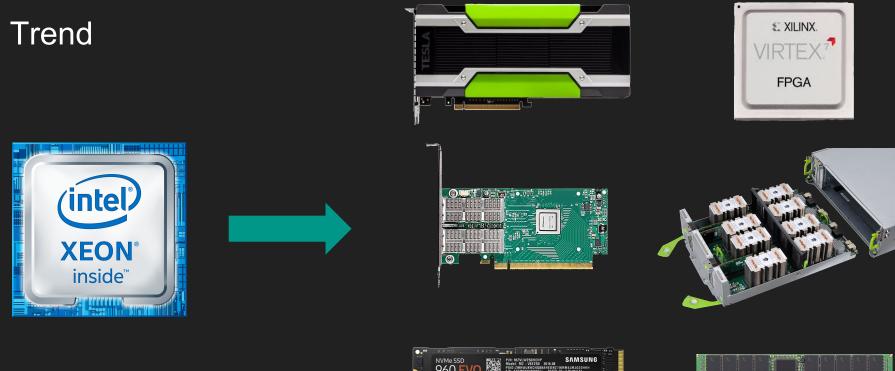
## Myself

- High performance trading systems

   Lock-free algos, distributed systems
- H2O
  - Distributed CPU machine learning, async SGD
- Flickr
  - Parallel deep learning Multi-GPU Caffe
  - Distributed deep learning CaffeOnSpark, RDMA, multicast, Hogwild
- UC Berkeley
  - NCCL Caffe, GPU cluster tooling
  - Bare Metal







**SD**<sup>®</sup>







#### ms software $\rightarrow$ µs hardware

Applications DBs, all server types,					
	System Libraries				
System Call Interface					
VFS			Sockets	Scheduler	
ext3/		ZFS	TCP/UDP		
LVM			IP	Virtual	
Block Device Interface			Ethernet	Memory	
Device Drivers					

Number crunching → GPU FS, block io  $\rightarrow$  Pmem Network stack -> RDMA RAID, replication  $\rightarrow$  Erasure codes Device mem  $\rightarrow$  Coherent fabrics And more: Video, crypto etc.



### Good ol' OS abstractions replaced by

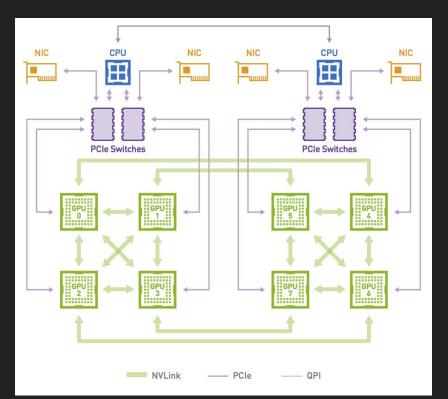
- CUDA
- OFED
- Libpmem
- DPDK
- SPDK
- Libfabric
- UCX
- VMA
- More every week...

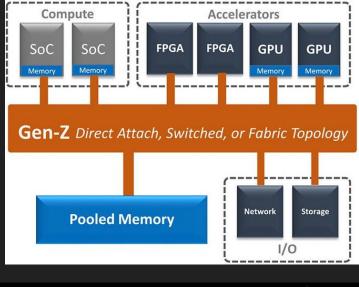
Faster, more powerful, but

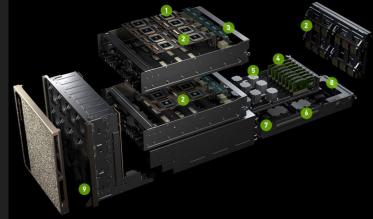
- More complex
- Non-interoperable
  - Namespaces
  - Security models
  - Failure models



#### Device-to-device flows







### Summary So Far - Big changes coming!

- CPU should orchestrate
  - Not in critical path
  - Device-to-device flows data and control







### What do we want?

- Stop over-specifying!
  - Location transparency: device-to-device, HA
  - Single namespace, security and failure models
- Simple model, slight extension of something familiar
- Forward and backward version compatible
- Thin efficient abstraction over hardware
- Stretch goals
  - Versioned git style
  - Capability system
  - Stateless apps, all state in versioned namespace



### Proposal

- Single namespace
  - File system like, distributed
  - HA using hardware erasure codes
- Nodes are data, compute steps or devices
  - E.g. numpy, protobuf, CUDA kernel, compute graph
  - Node execution starts when inputs are ready
- Versioned
  - Branch abstraction
  - Atomic merge or abort



#### API: mmap

- Extension to classic mmap
  - Distributed
  - Typed numpy, protobuf, other formats planned
- Python example

```
test = Test()
bm.mmap('/test', test)
i = test.field()
```



#### API: task

```
@bm.task
def compute(x, y):
    return x * y
```

```
# Runs locally
compute(1, 2)
```

```
# Might be rebalanced on cluster
data = bm.list()
bm.mmap("/data", data)
compute(data, 2)
```





#### API: branch

```
bm.branch = 'my_branch'
# ... modify dataset
bm.commit()
```

account1 = bm.mmap('/account1', my\_model.account())
account2 = bm.mmap('/account2', my\_model.account())
with bm.branch() as b:
 account1.set\_balance(account1.balance() + 12)
 account2.set\_balance(account2.balance() - 12)





### Internals

- Hardware erasure codes for all state
- Device-to-device transfers e.g. GPU Direct
- Device-to-device control e.g. GPU Direct Async
- Work stealing RDMA atomics
- Branches and lattice simplify a lot
  - No locks or coordination for most tasks
  - Atomicity simplifies consistency
    - Replaces transactions, e.g. KV, queues, persistent memory
    - No file system fsync, msync (Very hard! Rajimwale et al. DSN '11)
  - Allows duplicate work merge
  - Generalized staging / production split



### Data pull requests

- A data PR would contain
  - Data inputs
  - Code, compute graphs

Git Flow Chart

- Execution logs
- Single history tree
  - $\circ$  Code
  - Compilations
  - Executions
  - Data
- Persistent Jupyter Notebook?

Contras			
Î	FEATURE/JRA-1234 Working on JRA-1234	Remie Bolte 23:02:08 today 49a339b	
	MASTER 1.2.0 Merge branch 'release/1.2.0'	Remie Bolte 22:38:11 today a38b6ec	
0	DEVELOP RELEASE/1.3.0 Merge branch 'release/1.2.0' into develop	Remie Bolte 22:38:11 today 07bf32f	
	Merge branch 'master' into release/1.2.0	Remie Bolte 22:36:54 today 4b464c4	
0	Merge branch 'hotfix/1.1.3' into develop	Remie Bolte 22:36:41 today 8958747	
	1.1.3 Merge branch 'hotfix/1.1.3'	Remie Bolte 22:36:40 today aee21f0	
0	Adding a feature specific file	Remie Bolte 22:34:46 today 3e8f9d1	
	Merge branch 'hotfix/1.1.3' into release/1.2.0	Remie Bolte 22:33:55 today 55fb1b2	
	Removing obsolete file	Remie Bolte 22:33:39 today a93a486	
	Adding release notes	Remie Bolte 22:33:08 today b1b1d85	
	SIGNED_TAG Copy C.zip as D.zip	「 Anna Buttfield 四 2011-02-03 0a943a2	
	Add binary file C.zip	Anna Buttfield 🖾 2011-02-03 e0a3f6d	





Will be open sourced BSD

Contact me if interested - cyprien.noel@berkeley.edu



Thanks to our sponsor



