Probing Portable Performance of Parallel I/O Paradigms with MACSio

COE Performance Portability Meeting, Glendale, AZ.

Mark C Miller

April 21, 2016

LLNL-PRES-687907
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

What is MACSio?

- **M**ulti-purpose → Highly Flexible
- **A**pplication-**C**entric → High Level of Abstraction (LOA)
- **S**calable **i/o** → Parallel I/O "Workloads"
- Proxy application → More than a "benchmark"

MACSio aims to put the "Max"
In Scalable I/O Solutions

Why MACSio?

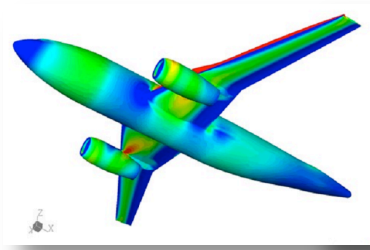
- Existing tools don't operate at LOA of real applications
- Benchmarks often limited in scope
  - Unable to drive wide a variety of I/O loads
  - Specific to a particular parallel I/O paradigm
  - Not easily extendible
- Application kernels also limited
  - Typically for compute so I/O ignored entirely
  - Only one specific I/O load
  - Only one specific I/O paradigm
  - Not easily extendible

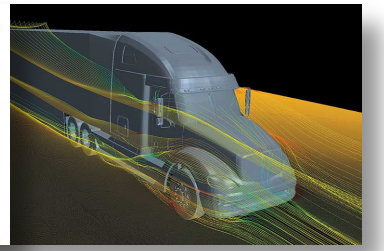
MACSio aims to be
Highly Flexible

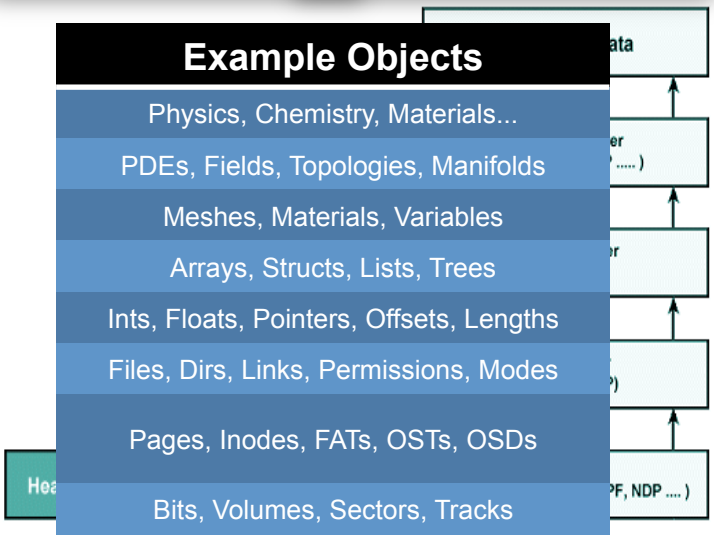
I/O, Abstraction and the HPC I/O "Stack"

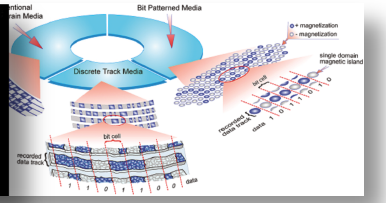
```
unsigned char img[256][256];
int mysz=sizeof(img)/nranks, off=mysz*rank;
struct hdr_t {int nx, ny, type, fill; } hdr;
int hdrsz = sizeof(hdr);

/* open image database and write object */
int x0 = 0, x1 = 256, y0 = off, y1 = off+mysz;
ImgDb = ImageDB_Open("my_images"):
ImageDB_Put("foo", x0,y0,x1,y1, IDB_UCHAR, fillval, img);
ImageDB_Close(Imgdb);
```

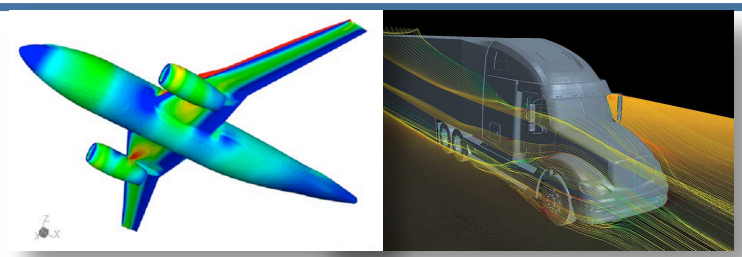








MACSio Operates at a Higher LOA

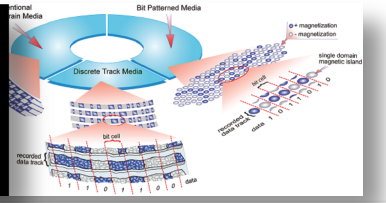


Increasing LOA

MACSio

Existing Tools

Concepts	Objects	Implementations
Real World Phenomena	Physics, Chemistry, Materials...	ALE3D, Albany, LAMPS
Continuous Mathematics	PDEs, Fields, Topologies, Manifolds	MFEM, FiberBundles, Sheafs, SAF
Discrete Numerical Models	Meshes, Materials, Variables	Silo, LibMesh, Exodus, ITAPS, VTK
Prog. Language Data Constructs	Arrays, Structs, Lists, Trees	HDF5, netCDF, ArrayIO, PDB
Primary Storage (Main Mem.)	Ints, Floats, Pointers, Offsets, Lengths	MPI-IO, XDR, stdio, aio, mmap
File System	Files, Dirs, Links, Permissions, Modes	POSIX IO
Secondary Storage (logical)	Pages, Inodes, FATs, OSTs, OSDs	GPFS, HDFS, Lustre / ext2/3, zfs, xfs, hfs
Secondary Storage (physical)	Bits, Volumes, Sectors, Tracks	hd, sd, cd, dvd, tape, ramdisk, flash



Evaluation of Existing Benchmarks

	Lang	LOA	Pmode	Coll	Real Data	Abs Keep	DIT	EZ Extend			Dir Ops	Perf DB	Last Active
								Pmode	Lib	MPI+			
IOR	C	1D Array	SSF, MIF	Yes	No	No	Some	No	Yes	?	Yes	No	13
FLASH-IO	Mix	AMR Mesh	SSF	No	No	Yes	No	No	No	No	No	No	11
S3D-IO	Fort	3D Array	SSF	Yes	No	Yes	No	No	No	No	No	No	?
BTIO	Fort	3D Array	SSF	Yes	Yes	Yes	No	No	No	No	No	No	03
GCRM-IO	C	AMR Mesh	SSF	Only	No	Yes	No	No	No	No	No	No	13
HACC-IO	C++	1D Array	SSF	No	No	No	No	Yes	Yes	?	No	No	12
b_eff_io	C	Filesystem	SSF, FPP	Yes	No	No	No	Yes	No	Yes	No	No	01
MPI Tile IO	C	Filesystem	SSF	Yes	No	No	Some	No	No	?	No	No	01
Non contig IO	C	Filesystem	SSF	Yes	No	No	No	No	No	Yes	No	No	05
fs_test	C	Filesystem	SSF	Yes	No	No	No	No	No	Yes	Yes	Yes	08
Parkbench	Fort	Filesystem	SSF	Yes	No	Yes	Some	No	No	No	No	No	97
Mdtest	C	Fileystem	None	No	No	No	No	No	No	?	Yes	No	03
Rompio	C	Filesystem	SSF	Yes	No	No	No	No	No	?	No	No	07
Tiobench	C	Filesystem	None				No						00

- LOA=Level of Abstraction
- PMode=Parallel Mode
- Coll=Collective I/Os
- Abs Keep=Abstraction Preserving?

- DIT=Data in transit services
  - precision, cksum, compress, subset...
- EZ-Extend (how hard to add)
  - Parallel Mode, I/O library, X in MPI+X
- Dir & FS ops
  - fstat, opendir, readdir, unlink
- Perf DB=Performance Database

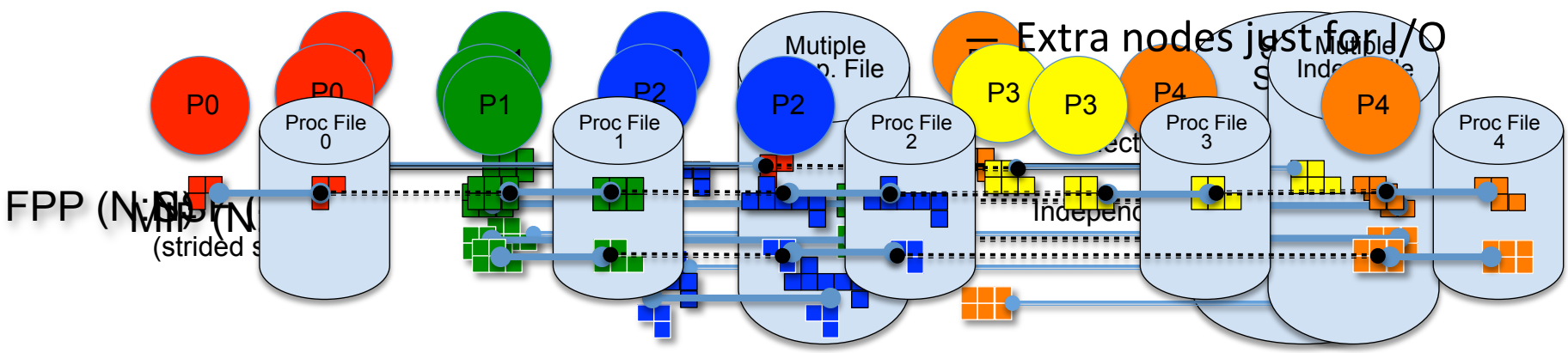
Flexibility in Parallel I/O Paradigms

File Per Processor (FPP)

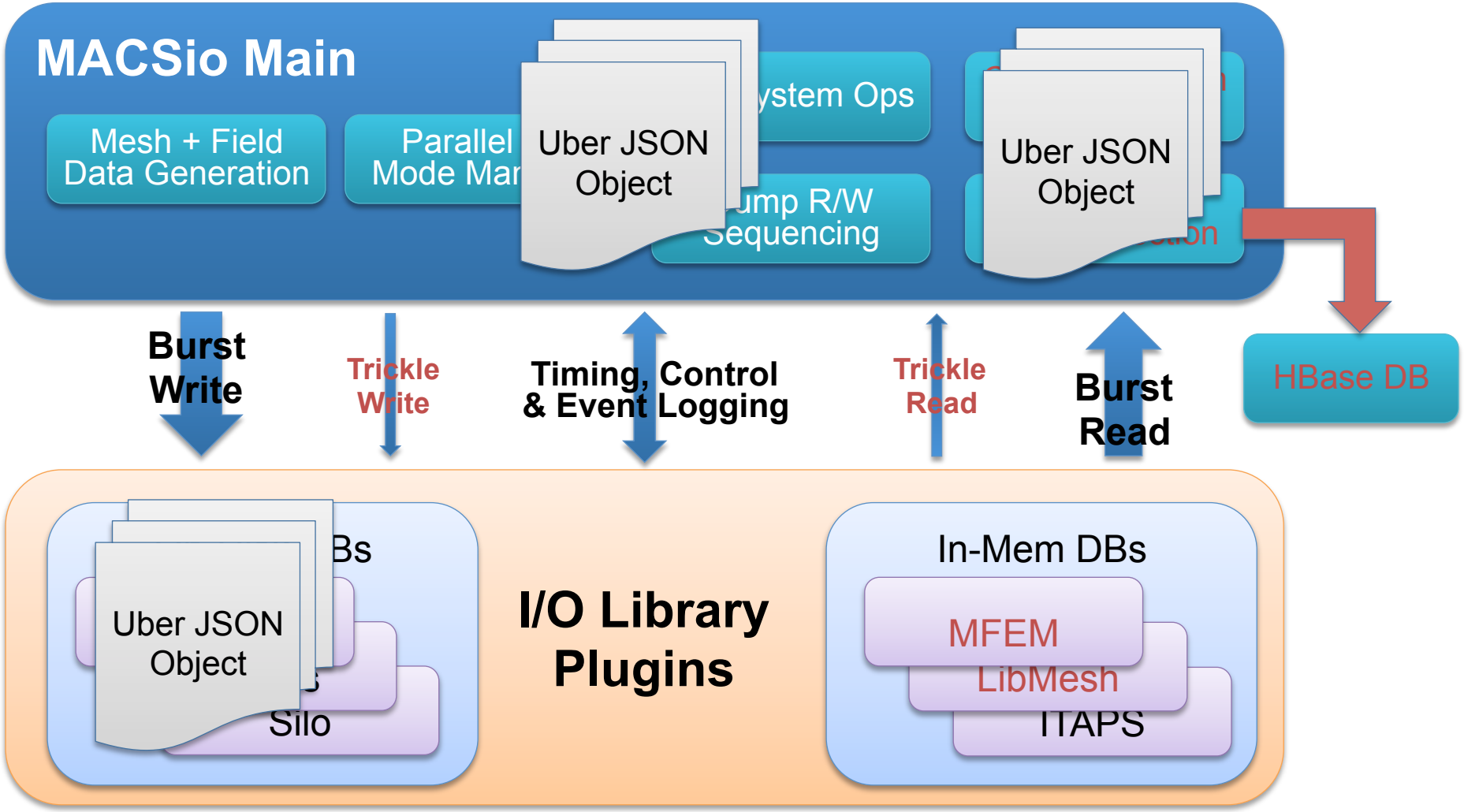
Multiple Independent File (MIF)

Single Shared File (SSF)

- Variations
  - Burst Buffers
  - Collective vs. Independent
  - Segmented vs. Strided
  - Two Phase
  - I/O node affinities
  - Custom I/O node code
  - Extra nodes just for I/O



MACSio Architecture



MACSio's Main Data Object

```
"Mesh": {
   "MeshType": "rectilinear",
   "ChunkID": 0, # unique per part
   "GeomDim": 2,
   "TopoDim": 2,
   "LogDims": [50,100],
   "Bounds": [0,0,0,1,1,0],
   "Coords": {
     "CoordBasis": "X,Y,Z",
     "XAxisCoords": [...],
     "YAxisCoords": [...]
   },
   "Topology": {
     "Type": "Templated",
     "DomainDim": 2,
     "RangeDim": 0,
     "ElemType": "Quad4",
     "Template": [0,1,51,50]
   }
 },
```

```
  "Vars": [
    {...},
    {...},
    {
      "name": "spherical",
      "centering": "zone",
      "data": [...]
    },
    {...},
    {...},
    {...}
  ],
  "GlobalLogIndices": [...],
  "GlobalLogOrigin": [...]
}
```

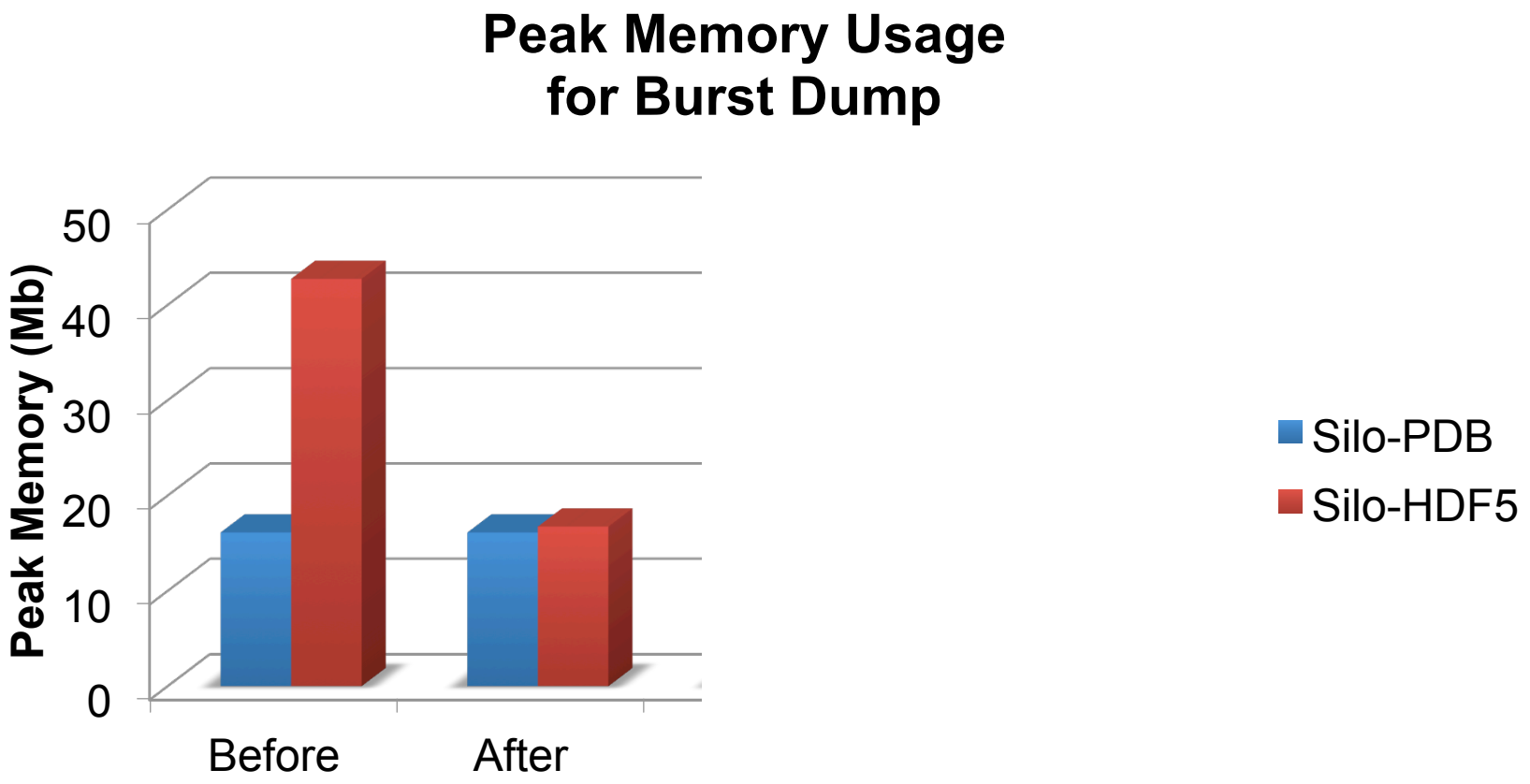
An Uber JSON-C Object

MACSio Enables us to Easily and Independently Test. . .

- Variety of Scaling Studies
- Variety of DIT-relevant data (not all zeros or random)
- Variety of I/O libraries
- Variety of I/O Paradigms
- Validate Data Written / Read
- Variety of DIT services
- Variety & mix of I/O loads
- Time & Space Performance
- Variety of other tools

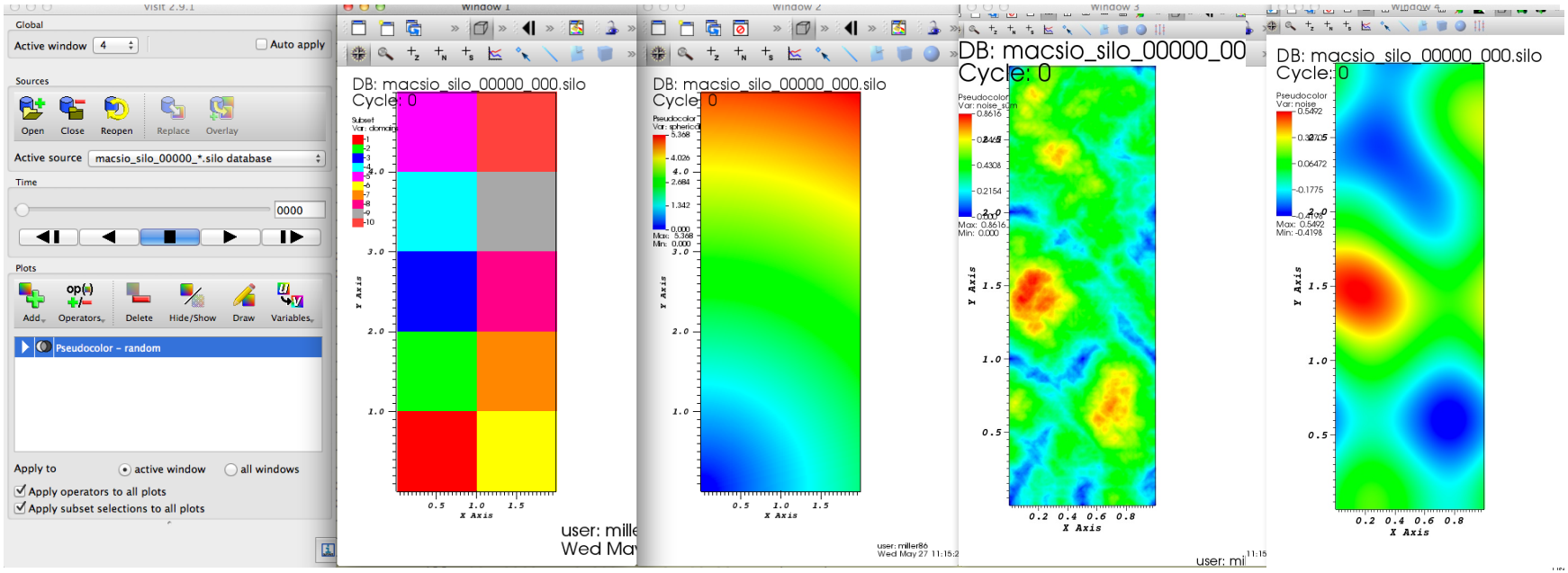
- Mesh part size & # parts/rank
- Mesh Type, Variable exprs + noise, Vary size, shape across ranks
- Plugins: HDF5, Silo, Exodus...
- SSF, MIF, FPP, coll./indep.
- Use plugin reader + cksums
- Prec. conv., compress, cksum, etc.
- Burst / trickle dumps & sequencing
- Timing, Logging, Memck classes
- SCR, VisIt, Darshan, Caliper

Recent Success: Identified & Improved Single-Node Peak Memory Usage



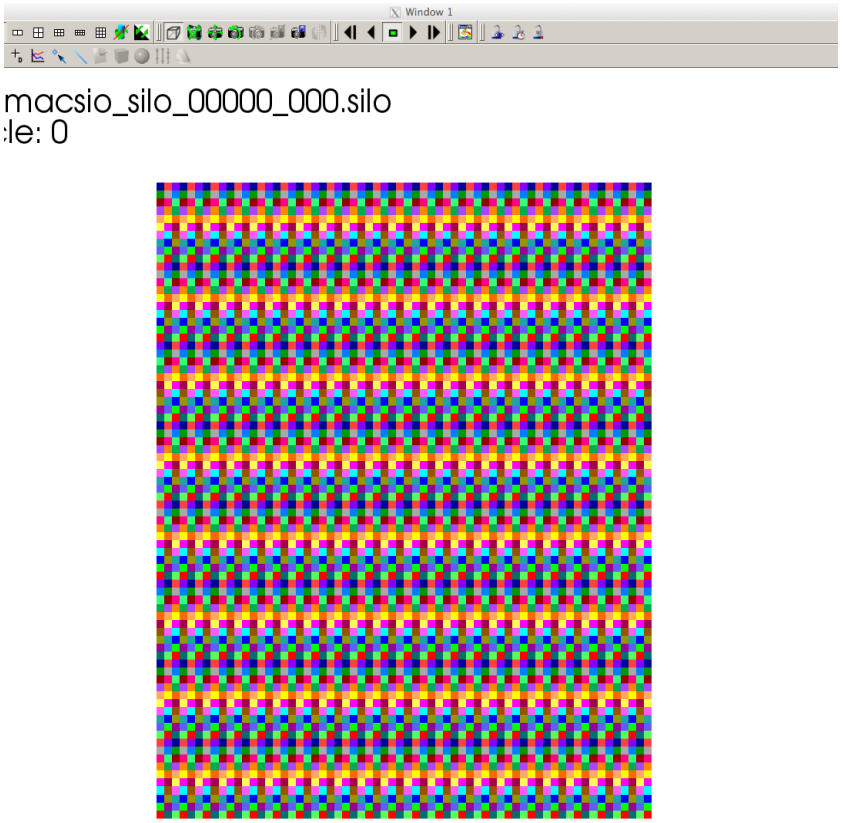
Placed $120K Contract with The HDF Group
To Further Improve Peak Memory by end of FY16

MACSio Examples (Silo) Visualized with VisIt



- 4 processors, avg_num_parts = 2.5, tot = 10 parts
- Showing block decomp, spherical var + 2 vars w/Perlin noise

2048 Core Example on Vulcan



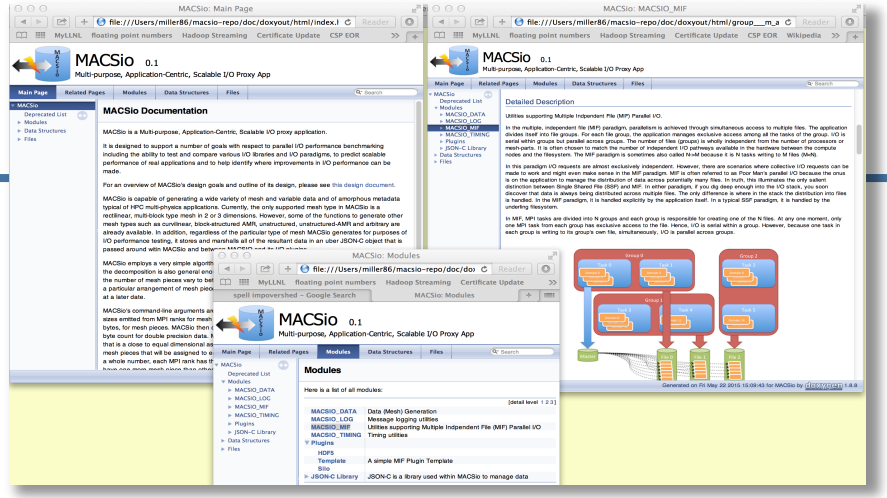
2048 Core @ 2.5 parts per core (vulcan)

user: miller86
Wed May 27 14:11:37 2

2048 Core @ 2.5 parts per core (vulcan)

user: miller86
Wed May 27 14:12:24 2

Getting MACSio



- Currently all C and uses GMake
  - MACSio main + utils: ~4500 lines
  - Silo: ~900, HDF5: ~600, Exodus: ~600
- Doxygen documented
- GPL Open Source, on Github
  - https://github.com/LLNL/MACSio
- Volunteers for new plugins?
  - AWE added a TyphonIO plugin
  - Contact: miller86@llnl.gov

Lawrence Livermore
National Laboratory

Files are so four decades ago...

- "POSIX IO Must Die!"
  - Jeffery Layton, Linux Magazine, March 2010
  - Oft used rational for replacing file systems with object stores
- POSIX File Semantics
  - Sequential consistency, Atomicity, ...
  - Locking used to implement these hurts!
- POSIX File Interfaces
  - `stat, open, read, write, seek, close`
  - Our workflows operate everywhere, Laptop to LCF, on these interfaces
  - Ginormous effort to replace!

By all means, let the semantics die!
But interfaces and files are here to stay

Easily and Independently Vary Test Parameters For...

- Weak / Strong Scaling Studies
- I/O Hardware Architecture
  - Burst Buffers
  - I/O & Compute Node Affinities
  - File system interfaces
- Different I/O libraries (plugins)
  - Silo, Exodus
  - HDF5, Xdmf, Adios...
- Data-in-transit (DIT) services
  - transpose / re-order
  - compression
  - checksumming
  - precision conversion
- Parallel I/O paradigms
  - N:1, N:M, N:N
  - Strided/Segmented,
  - Collective/Independent
  - Two-Phase
- Application Data Objects
  - Structured, Unstructured, AMR, Arbitrary Polyhedral meshes
  - Scalar, vector, tensor variables
  - Mixing Materials and subset vars
- Integrate with external tools
  - SCR
  - Darshan
  - VisIt / Paraview