E4S: Extreme-scale Scientific Software Stack

High Throughput Computing (HTC23)
https://agenda.hep.wisc.edu/event/2014/contributions/28449/
July 14, 2023, 11:15am CT - 11:45am CT
Howard Auditorium
Fluno Center on the University of Wisconsin-Madison Campus

Prof. Sameer Shende
Research Professor and Director,
Performance Research Laboratory, OACISS, University of Oregon
President and Director, ParaTools, Inc.
https://e4s.io/talks/E4S_HTC23.pdf
Challenges

- As our software gets more complex, it is getting harder to install our tools and libraries correctly in an integrated and interoperable software stack to deploy our applications to the high throughput computing (HTC) platforms!
- E4S is a containerized platform that features a capable HTC software stack with support for GPUs
Extreme-scale Scientific Software Stack (E4S)
Exascale Computing Project (ECP)

https://exascaleproject.org
ECP’s holistic approach uses co-design and integration to achieve exascale computing

**Application Development (AD)**
- Develop and enhance the predictive capability of applications critical to DOE
- **24 applications**
  - National security, energy, Earth systems, economic security, materials, data
- **6 Co-Design Centers**
  - Machine learning, graph analytics, mesh refinement, PDE discretization, particles, online data analytics

**Software Technology (ST)**
- Deliver expanded and vertically integrated software stack to achieve full potential of exascale computing
- **71 unique software products** spanning programming models and run times, math libraries, data and visualization

**Hardware and Integration (HI)**
- Integrated delivery of ECP products on targeted systems at leading DOE HPC facilities
- **6 US HPC vendors** focused on exascale node and system design; application integration and software deployment to Facilities

**Performant mission and science applications at scale**
US DOE HPC Roadmap to Exascale Systems

2012-2020

- Titan
  ORNL
  Cray/AMD/NVIDIA
- Mira
  ANL
  IBM BG/Q
- Theta
  ANL
  Cray/Intel KNL
- Cori
  LBNL
  Cray/Intel Xeon/KNL
- Sequoia
  LLNL
  IBM BG/Q
- Trinity
  LANL/SNL
  Cray/Intel Xeon/KNL
- Sierra
  LLNL
  IBM/NVIDIA
- Summit
  ORNL
  IBM/NVIDIA

2021-2023

- Frontier
  ORNL
  HPE/AMD
- Aurora
  ANL
  Intel/HPE
- Perlmutter
  LBNL
  HPE/AMD/NVIDIA
- CROSSROADS
  LANL/SNL
  HPE/Intel
- El Capitan
  LLNL
  HPE/AMD
- Sierra
  LLNL
  IBM/NVIDIA

Exascale Systems
Goal
Build a comprehensive, coherent software stack that enables application developers to productively develop highly parallel applications that effectively target diverse exascale architectures

- Prepare SW stack for scalability with massive on-node parallelism
- Extend existing capabilities when possible, develop new when not
- Guide, and complement, and integrate with vendor efforts
- Develop and deliver high-quality and robust software products
Extreme-scale Scientific Software Stack (E4S)

- **E4S**: HPC Software Ecosystem – a curated software portfolio
- A Spack-based distribution of software tested for interoperability and portability to multiple architectures with support for GPUs from NVIDIA, AMD, and Intel in each release
- Available from source, containers, cloud, binary caches
- Leverages and enhances SDK interoperability thrust
- Not a commercial product – an open resource for all
- Oct 2018: E4S 0.1 - 24 full, 24 partial release products
- Jan 2019: E4S 0.2 - 37 full, 10 partial release products
- Nov 2019: E4S 1.0 - 50 full, 5 partial release products
- Feb 2020: E4S 1.1 - 61 full release products
- Nov 2020: E4S 1.2 (aka, 20.10) - 67 full release products
- Feb 2021: E4S 21.02 - 67 full release, 4 partial release
- May 2021: E4S 21.05 - 76 full release products
- Aug 2021: E4S 21.08 - 88 full release products
- Nov 2021: E4S 21.11 - 91 full release products
- Feb 2022: E4S 22.02 – 100 full release products
- May 2022: E4S 22.05 – 101 full release products
- August 2022: E4S 22.08 – 102 full release products
- November 2022: E4S 22.11 – 103 full release products
- February 2023: E4S 23.02 – 106 full release products
- May 2023: E4S 23.05 – 109 full release products

Also include other products e.g.,
AI: PyTorch, TensorFlow (CUDA, ROCm)
Co-Design: AMReX, Cabana, MFEM
EDA: Xyce
E4S: Extreme-scale Scientific Software Stack

- E4S is a community effort to provide open-source software packages for developing, deploying and running scientific applications on HPC platforms.
- E4S has built a comprehensive, coherent software stack that enables application developers to productively develop highly parallel applications that effectively target diverse exascale architectures.
- E4S provides a curated, Spack based software distribution of 100+ HPC, 50+ EDA (e.g., Xyce), and AI/ML packages (e.g., TensorFlow, PyTorch).
- With E4S Spack binary build caches, E4S supports both bare-metal and containerized deployment for GPU based platforms.
  - X86_64, ppc64le (IBM Power 9), aarch64 (ARM64) with support for GPUs from NVIDIA, AMD, and Intel
  - HPC and AI/ML packages are optimized for GPUs and CPUs.
- Container images on DockerHub and E4S website of pre-built binaries of ECP ST products.
- Base images and full featured containers (with GPU support).
- Commercial support for E4S through ParaTools, Inc. for installation, maintaining an issue tracker, and ECP AD engagement.
- e4s-cl container launch tool allows binary distribution of applications by substituting MPI in the containerized app with the system MPI. e4s-alc is a tool to create custom container images from base images.
- Quarterly releases: E4S 23.05 released on May 31, 2023: https://e4s.io/talks/E4S_23.05.pdf
- E4S for commercial cloud platforms: AWS image supports MPI implementations and containers with remote desktop (DCV).
  - Intel MPI, NVHPC, MVAPICH2, MPICH, MPC, OpenMPI

https://e4s.io
Getting started with E4S on OSG-HTC

```bash
[sameer@ap20 example]$ ls /cvmfs/singularity.opensciencegrid.org/ecpe4s/
  e4s-cuda:22.11  e4s-cuda:23.05  e4s-oneapi:22.11  e4s-oneapi:23.05  e4s-rocm:22.11  e4s-rocm:23.05

[sameer@ap20 example]$ cat workload.submit
+SingularityImage = "/cvmfs/singularity.opensciencegrid.org/ecpe4s/e4s-cuda:23.05"
requirements = GPUs:Capability >= 8.0
executable  = test.sh
arguments   =
input       =
output      = $(Cluster).$(Process).out
error       = $(Cluster).$(Process).err
log         = $(Cluster).$(Process).log
request_cpus = 1
request_gpus = 1
request_memory = 8GB
queue 1

[sameer@ap20 example]$ cat test.sh
#!/bin/bash
set -e
echo "I'm running on "'hostname -f'
echo "OSG site: $OSG_SITE_NAME"

echo
export CUDA_VISIBLE_DEVICES=0
cd /opt/demo/python_tests
./run.sh

echo "Arch list:" python -c "import torch; print(torch.cuda.get_arch_list())"
nvidia-smi
lscpu

[sameer@ap20 example]$ condor_submit workload.submit
```

cd /cvmfs/singularity.opensciencegrid.org/ecpe4s
I'm running on gpu10.cluster
OSG site: PDX-Coeus-CE1
Fri 6 May
PyTorch DeviceName: NVIDIA RTX A5000
99 851.84238701171875
199 14.990115421906133
299 0.35356718368538273
399 0.000744419636432194
499 0.0004735153649330139
Arch list: ['sm_35', 'sm_70', 'sm_75', 'sm_80', 'sm_90']
Sat Jul 8 09:36:17 2023

<table>
<thead>
<tr>
<th>NVIDIA-SMI 530.30.02</th>
<th>Driver Version: 530.30.02</th>
<th>CUDA Version: 12.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPU Name</td>
<td>Persistence-M</td>
<td>Bus-ID</td>
</tr>
<tr>
<td>Fan</td>
<td>Temp</td>
<td>Perf</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>0 NVIDIA RTX A5000</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>38%</td>
<td>3IC</td>
<td>P8</td>
</tr>
<tr>
<td>NVIDIA RTX A5000</td>
<td>On</td>
<td>00000000:01:00.0 Off</td>
</tr>
<tr>
<td>1 NVIDIA RTX A5000</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>38%</td>
<td>24C</td>
<td>P8</td>
</tr>
<tr>
<td>NVIDIA RTX A5000</td>
<td>Off</td>
<td>00000000:01:00.0 Off</td>
</tr>
<tr>
<td>2 NVIDIA RTX A5000</td>
<td>On</td>
<td>00000000:01:00.0 Off</td>
</tr>
<tr>
<td>38%</td>
<td>24C</td>
<td>P8</td>
</tr>
<tr>
<td>NVIDIA RTX A5000</td>
<td>Off</td>
<td>00000000:01:00.0 Off</td>
</tr>
<tr>
<td>3 NVIDIA RTX A5000</td>
<td>Off</td>
<td>Off</td>
</tr>
<tr>
<td>38%</td>
<td>24C</td>
<td>P8</td>
</tr>
</tbody>
</table>

Processes:
<table>
<thead>
<tr>
<th>GPU</th>
<th>CI</th>
<th>PID</th>
<th>Type</th>
<th>Process name</th>
<th>GPU Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>ID</td>
<td>Usage</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No running processes found</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
Address sizes: 43 bits physical, 48 bits virtual
CPU(s): 64
On-line CPU(s) list: 0-63
E4S Support for AI/ML frameworks with V100, A100, and H100 GPUs

```python
Singularity> python
Python 3.8.10 (default, Nov 14 2022, 12:59:47)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy
>>> import scipy
>>> import matplotlib
>>> import tensorflow
>>> tensorflow.__version__
'2.12.0'
>>> import torch
>>> torch.__version__
'2.0.0'
>>> torch.cuda.get_device_name(torch.cuda.current_device())
'NVIDIA H100 PCIe'
```

E4S 23.05 supports NVIDIA H100 GPUs with TensorFlow 2.12.0 and PyTorch 2.0.0
Python tools in E4S 23.05 on Expanse at SDSC

cd /cm/shared/apps/containers/singularity/e4s/v23.05
E4S includes Julia with support for CUDA and MPI

```plaintext
Singularity> which julia
/usr/local/julia/1.9.0/bin/julia
Singularity> cd /opt/demo/Julia/
GrayScott.jl/ Mandelbrot/ MPI_example/
Singularity> cd /opt/demo/Julia/MPI_example/
Singularity> ls
foo.jl  run.sh
Singularity> cat foo.jl
# examples/01-hello.jl
using MPI
MPI.Init()

comm = MPI.COMM_WORLD
println("Hello world, I am $(MPI.Comm_rank(comm)) of $(MPI.Comm_size(comm))")
MPI.Barrier(comm)

# From: https://juliaparallel.org/MPI.jl/stable/usage/
Singularity> cat ./run.sh
#!/bin/bash
mpirun -np 4 julia --project ./foo.jl
Singularity> ./run.sh
Hello world, I am 0 of 4
Hello world, I am 1 of 4
Hello world, I am 2 of 4
Hello world, I am 3 of 4
```

Singularity>
Considerations while deploying HPC/AI workloads to the cloud

- Which cloud provider?
  - AWS, OCI, GCP, Azure, …
  - Why not all?

- HPC and AI/ML workloads need low latency, high bandwidth
  - Which MPI?

- Which image?
  - Base Ubuntu without HPC tools or libraries? Too steep a learning curve

- Provisioning and building the image on different cloud providers
  - Command line interfaces can be cumbersome to use

- Bursting to the cloud from on-prem clusters using batch submission scripts?
Key considerations for cloud-based deployment for E4S

• MPI - the core inter-node communication library has several implementations
  – Intel MPI, MVAPICH2-X, OpenMPI
  – Interfacing MPI with the job scheduling package (MOAB, Torque, SLURM)

• Cloud providers have different inter-node network adapters:
  – Elastic Fabric Adapter (EFA) on AWS
  – Infiniband on Azure
  – Mellanox Connect-X 5 Ethernet (ROCE) on Oracle Cloud Infrastructure (OCI)

• Intra-node communication with XPMEM (driver and kernel module support is critical)

• GPU Direct Async (GDR) support for communication between GPUs in MVPICH-Plus release

• ParaTools, Inc. building E4S optimized with MVAPICH-Plus for AWS, OCI, GCP, and Azure

• Using Adaptive Computing’s ODDC interface to launch E4S jobs on multiple cloud providers!
Adaptive Computing’s ODDC interface for E4S
E4S 23.05 AWS image: US-West2 (OR)

- Intel oneAPI
- CUDA
- NVHPC
- ROCm
- AWS DCV
- Spack Build Cache
- ECP: Nalu-Wind
- Trilinos 13.4.0
- OpenFOAM
- ParaView
- TAU
- Docker
- Shifter
- Charliecloud
- E4S Singularity
- EDA tools…
E4S for Commercial Cloud Platforms for EDA on AWS

- **E4S:** HPC Software Ecosystem – a curated software portfolio for Electronic Design Automation

  - Magic
  - ACT
  - Klayout
  - Qflow
  - Xschem
  - Xcircuit
  - Yosys
  - Volator
  - OpenROAD
  - OpenLane
  - OpenFASOC
  - iVerilog
  - Gtkwave
  - Irsim
  - Qrouter
  - Fault
  - GDS3D
  - Rggen

  - **Python tools**
    - Cocotb
    - Amaranth
    - Edalize
    - Gdsfactory
    - Gdsyre
    - Edalize
    - Gdsfactory
    - Gdspy
    - OpenRAM
    - Gdstk
    - Silicon compiler
    - Volare ...

  - **PDKs**
    - GF
    - Skywater

  - **E4S EDA on AWS**

  - [https://e4s.io/eda](https://e4s.io/eda)
E4S for Commercial Cloud Platforms for EDA on AWS

- E4S: HPC Software Ecosystem – a curated software portfolio for Electronic Design Automation
- OpenROAD

E4S EDA on AWS
- Magic
- ACT
- Klayout
- Qflow
- Xschem
- Xcircuit
- Yosys
- Volator
- OpenROAD
- OpenLane
- OpenFASOC
- iVerilog
- Gtkwave
- Irsim
- Qrouter
- Fault
- GDS3D
- Rggen

Python tools
- Cocotb
- Amaranth
- Edalize
- Gdsfactory
- Gdsipy
- OpenRAM
- Gdstk
- Silicon compiler
- Volare ...

PDKs
- GF
- Skywater
### E4S for Commercial Cloud Platforms for EDA on AWS

**E4S**: HPC Software Ecosystem – a curated software portfolio for Electronic Design Automation

<table>
<thead>
<tr>
<th>#</th>
<th>Packages currently in E4S</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Magic</td>
<td><a href="http://opencircuitdesign.com/magic/">http://opencircuitdesign.com/magic/</a></td>
</tr>
<tr>
<td>2</td>
<td>Xyce</td>
<td><a href="https://xyce.sandia.gov">https://xyce.sandia.gov</a></td>
</tr>
<tr>
<td>3</td>
<td>NGSPICE</td>
<td><a href="https://ngspice.sourceforge.io">https://ngspice.sourceforge.io</a></td>
</tr>
<tr>
<td>4</td>
<td>KLayout</td>
<td><a href="https://www.klayout.de">https://www.klayout.de</a></td>
</tr>
<tr>
<td>5</td>
<td>Qflow</td>
<td><a href="http://opencircuitdesign.com/qflow">http://opencircuitdesign.com/qflow</a></td>
</tr>
<tr>
<td>6</td>
<td>OR-Tools</td>
<td><a href="https://developers.google.com/optimization">https://developers.google.com/optimization</a></td>
</tr>
<tr>
<td>7</td>
<td>IRSIM</td>
<td><a href="http://opencircuitdesign.com/irsim/">http://opencircuitdesign.com/irsim/</a></td>
</tr>
<tr>
<td>8</td>
<td>OpenROAD</td>
<td><a href="https://github.com/The-OpenROAD-Project/OpenROAD">https://github.com/The-OpenROAD-Project/OpenROAD</a></td>
</tr>
<tr>
<td>9</td>
<td>OpenLane</td>
<td><a href="https://openlane.readthedocs.io/">https://openlane.readthedocs.io/</a></td>
</tr>
<tr>
<td>10</td>
<td>OpenFASOC</td>
<td><a href="https://openfasoc.readthedocs.io/">https://openfasoc.readthedocs.io/</a></td>
</tr>
<tr>
<td>11</td>
<td>Open_PDKs</td>
<td><a href="http://opencircuitdesign.com/open_pdkss/">http://opencircuitdesign.com/open_pdkss/</a></td>
</tr>
<tr>
<td>12</td>
<td>Netgen</td>
<td><a href="http://opencircuitdesign.com/netgen/">http://opencircuitdesign.com/netgen/</a></td>
</tr>
<tr>
<td>13</td>
<td>Yosys</td>
<td><a href="https://github.com/YosysHQ/yosys">https://github.com/YosysHQ/yosys</a></td>
</tr>
<tr>
<td>14</td>
<td>Xcircuit</td>
<td><a href="http://opencircuitdesign.com/xcircuit/">http://opencircuitdesign.com/xcircuit/</a></td>
</tr>
<tr>
<td>15</td>
<td>Graywolf</td>
<td><a href="https://github.com/rubund/graywolf">https://github.com/rubund/graywolf</a></td>
</tr>
<tr>
<td>16</td>
<td>OpenSTA</td>
<td><a href="https://github.com/The-OpenROAD-Project/OpenSTA">https://github.com/The-OpenROAD-Project/OpenSTA</a></td>
</tr>
<tr>
<td>17</td>
<td>OpenTimer</td>
<td><a href="https://github.com/OpenTimer/OpenTimer">https://github.com/OpenTimer/OpenTimer</a></td>
</tr>
<tr>
<td>18</td>
<td>Qrouter</td>
<td><a href="http://opencircuitdesign.com/qrouter/">http://opencircuitdesign.com/qrouter/</a></td>
</tr>
<tr>
<td>19</td>
<td>Xscheme</td>
<td><a href="https://github.com/silicon-vlsi-org/eda-xschem">https://github.com/silicon-vlsi-org/eda-xschem</a></td>
</tr>
<tr>
<td>20</td>
<td>RISC-V GNU Toolchain</td>
<td><a href="https://github.com/riscv-collab/riscv-gnu-toolchain">https://github.com/riscv-collab/riscv-gnu-toolchain</a></td>
</tr>
<tr>
<td>21</td>
<td>Fault: Design for Test</td>
<td><a href="https://github.com/AUCOHL/Fault">https://github.com/AUCOHL/Fault</a></td>
</tr>
<tr>
<td>22</td>
<td>NVC</td>
<td><a href="https://github.com/nickg/nvc">https://github.com/nickg/nvc</a></td>
</tr>
<tr>
<td>23</td>
<td>Amaranth</td>
<td><a href="https://github.com/amaranth-lang/amaranth">https://github.com/amaranth-lang/amaranth</a></td>
</tr>
<tr>
<td>24</td>
<td>Cocotb</td>
<td><a href="https://github.com/cocotb/cocotb">https://github.com/cocotb/cocotb</a></td>
</tr>
</tbody>
</table>

https://e4s.io/eda
E4S for Commercial Cloud Platforms for EDA on AWS

- **E4S**: HPC Software Ecosystem – a curated software portfolio for Electronic Design Automation

<table>
<thead>
<tr>
<th>#</th>
<th>Packages currently in E4S</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>37</td>
<td>Padring</td>
<td><a href="https://github.com/donn/padring">https://github.com/donn/padring</a></td>
</tr>
<tr>
<td>38</td>
<td>Pyverilog</td>
<td><a href="https://github.com/PyHDI/Pyverilog">https://github.com/PyHDI/Pyverilog</a></td>
</tr>
<tr>
<td>39</td>
<td>OpenRAM</td>
<td><a href="https://github.com/VLSIDA/OpenRAM">https://github.com/VLSIDA/OpenRAM</a></td>
</tr>
<tr>
<td>40</td>
<td>Rggen</td>
<td><a href="https://github.com/rpggen/rggen">https://github.com/rpggen/rggen</a></td>
</tr>
<tr>
<td>41</td>
<td>Spyci</td>
<td><a href="https://github.com/gmagno/spyci">https://github.com/gmagno/spyci</a></td>
</tr>
<tr>
<td>42</td>
<td>Volare</td>
<td><a href="https://github.com/efabless/volare">https://github.com/efabless/volare</a></td>
</tr>
<tr>
<td>43</td>
<td>Siliconcompiler</td>
<td><a href="https://github.com/siliconcompiler/siliconcompiler">https://github.com/siliconcompiler/siliconcompiler</a></td>
</tr>
<tr>
<td>44</td>
<td>Verilator</td>
<td><a href="https://github.com/verilator/verilator">https://github.com/verilator/verilator</a></td>
</tr>
<tr>
<td>45</td>
<td>Sky130</td>
<td>SkyWater Technologies 130nm CMOS PDK</td>
</tr>
<tr>
<td>46</td>
<td>Actflow</td>
<td><a href="https://github.com/asyncvlsi/actflow.git">https://github.com/asyncvlsi/actflow.git</a></td>
</tr>
<tr>
<td>47</td>
<td>Qucs-s</td>
<td><a href="https://github.com/Qucs">https://github.com/Qucs</a></td>
</tr>
<tr>
<td>48</td>
<td>ADMS</td>
<td><a href="https://github.com/Qucs/ADMS.git">https://github.com/Qucs/ADMS.git</a></td>
</tr>
<tr>
<td>49</td>
<td>Gdstk</td>
<td><a href="https://heitzmann.github.io/gdstk/">https://heitzmann.github.io/gdstk/</a></td>
</tr>
<tr>
<td>50</td>
<td>xcell</td>
<td><a href="https://github.com/asyncvlsi/xcell.git">https://github.com/asyncvlsi/xcell.git</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Packages currently in E4S</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>25</td>
<td>Covered</td>
<td><a href="https://github.com/hpretl/verilog-covered">https://github.com/hpretl/verilog-covered</a></td>
</tr>
<tr>
<td>26</td>
<td>Edalize</td>
<td><a href="https://github.com/olofk/edalize">https://github.com/olofk/edalize</a></td>
</tr>
<tr>
<td>27</td>
<td>Gav3-xschem</td>
<td><a href="https://github.com/StefanSchippers/xschem-gaw.git">https://github.com/StefanSchippers/xschem-gaw.git</a></td>
</tr>
<tr>
<td>28</td>
<td>GDSFactory</td>
<td><a href="https://github.com/gdsfactory/gdsfactory">https://github.com/gdsfactory/gdsfactory</a></td>
</tr>
<tr>
<td>29</td>
<td>GDSPy</td>
<td><a href="https://github.com/heitzmann/gdspy">https://github.com/heitzmann/gdspy</a></td>
</tr>
<tr>
<td>30</td>
<td>GDS3D</td>
<td><a href="https://github.com/trilomix/GDS3D">https://github.com/trilomix/GDS3D</a></td>
</tr>
<tr>
<td>31</td>
<td>Ghdl</td>
<td><a href="https://github.com/ghdl/ghdl">https://github.com/ghdl/ghdl</a></td>
</tr>
<tr>
<td>32</td>
<td>Gtkwave</td>
<td><a href="https://github.com/gtkwave/gtkwave">https://github.com/gtkwave/gtkwave</a></td>
</tr>
<tr>
<td>33</td>
<td>iic-osic</td>
<td><a href="https://github.com/hpretl/iic-osic.git">https://github.com/hpretl/iic-osic.git</a></td>
</tr>
<tr>
<td>34</td>
<td>Iverilog</td>
<td><a href="https://github.com/steveicarus/iverilog.git">https://github.com/steveicarus/iverilog.git</a></td>
</tr>
<tr>
<td>35</td>
<td>Netlistsvg</td>
<td><a href="https://github.com/nturley/netlistsvg">https://github.com/nturley/netlistsvg</a></td>
</tr>
<tr>
<td>36</td>
<td>Ngspyce</td>
<td><a href="https://github.com/ignamv/ngspyce">https://github.com/ignamv/ngspyce</a></td>
</tr>
</tbody>
</table>
Spack

- E4S uses the Spack package manager for software delivery
- Spack provides the ability to specify versions of software packages that are and are not interoperable.
- Spack is a build layer for not only E4S software, but also a large collection of software tools and libraries outside of ECP ST.
- Spack supports achieving and maintaining interoperability between ST software packages.
- [https://spack.io](https://spack.io)
Spack is a flexible package manager for HPC

• How to install Spack (works out of the box):

$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh

• How to install a package:

$ spack install tau

• TAU and its dependencies are installed within the Spack directory.

• Unlike typical package managers, Spack can also install many variants of the same build.
  – Different compilers
  – Different MPI implementations
  – Different build options
Spack provides the **spec** syntax to describe custom configurations

- Each expression is a **spec** for a particular configuration
  - Each clause adds a constraint to the spec
  - Constraints are optional – specify only what you need.
  - Customize install on the command line!

- Spec syntax is recursive
  - Full control over the combinatorial build space

```bash
$ git clone https://github.com/spack/spack
$ . spack/share/spack/setup-env.sh
$ spack compiler find # set up compilers
$ spack external find # set up external packages

$ spack install tau # unconstrained
$ spack install tau@2.32 @ custom version
$ spack install tau@2.32 %gcc@9.3.0 % custom compiler
$ spack install tau@2.32 %gcc@9.3.0 +rocm +/- build option
$ spack install tau@2.32 %gcc@9.3.0 +mpi ^mvapich2@2.3~wrapperrpath ^ dependency information
```
The Spack community is growing rapidly

- **Spack simplifies HPC software for:**
  - Users
  - Developers
  - Cluster installations
  - The largest HPC facilities

- **Spack is central to ECP’s software strategy**
  - Enable software reuse for developers and users
  - Allow the facilities to consume the entire ECP stack

- **The roadmap is packed with new features:**
  - Building the ECP software distribution
  - Better workflows for building containers
  - Stacks for facilities
  - Chains for rapid dev workflow
  - Optimized binaries
  - Better dependency resolution

Visit spack.io

github.com/spack/spack

@spackpm
E4S 23.05 container images now available!
See Downloads for more information on E4S 23.05.

What is E4S?

The Extreme-scale Scientific Software Stack (E4S) is a community effort to provide open source software packages for developing, deploying and running scientific applications on high-performance computing (HPC) platforms. E4S provides from-source builds and containers of a broad collection of HPC software packages.
Acquiring E4S Containers

The current E4S container offerings include Docker and Singularity images capable of running on X86_64, PPC64LE, and AARCH64 architectures. Our full E4S Release images are based on Ubuntu 20.04 (x86_64, aarch64, ppc64le). In addition to offering a full E4S image containing a comprehensive selection of E4S software released on a quarterly cycle, we also offer a set of minimal base images suitable for use in Continuous Integration (CI) pipelines where Spack is used to build packages.

Docker images are available on the E4S Docker Hub.

Please see the E4S 23.05 Release Notes.
Download E4S 23.05 GPU Container Images: NVIDIA, AMD, Intel

• Separate full featured Singularity images for 3 GPU architectures
• GPU full featured images for
  – x86_64 (Intel, AMD, NVIDIA)
  – ppc64le (NVIDIA)
  – aarch64 (NVIDIA)
• Full featured images available on Dockerhub
• 100+ products on 3 architectures

https://e4s.io
Download E4S 23.05 GPU Container Images: AMD, Intel, and NVIDIA

Note on Container Images

Container images contain binary versions of the Full Release packages listed above. Full-featured GPU-enabled container images are available from Dockerhub:

```sh
# docker pull ecpe4s/e4s-cuda:23.05
# docker pull ecpe4s/e4s-rocm:23.05
# docker pull ecpe4s/e4s-oneapi:23.05
```

E4S Full GPU Images

These images contain a full Stack-based deployment of E4S, including GPU-enabled packages for NVIDIA, AMD, or Intel GPUs.

These images also contain TensorFlow, PyTorch, and TAU.

AMD ROCm (x86_64)
- ecpe4s/e4s-rocm:23.05
- e4s-rocm90a-x86_64-23.05.sif (S) mirror 1
- e4s-rocm908-x86_64-23.05.sif (S) mirror 1

NVIDIA CUDA (x86_64, PPC64LE, AARCH64)
- ecpe4s/e4s-cuda:23.05
- e4s-cuda80-x86_64-23.05.sif (S) mirror 1
- e4s-cuda90-x86_64-23.05.sif (S) mirror 1
- e4s-cuda70-ppc64le-23.05.sif (S) mirror 1
- e4s-cuda75-aarch64-23.05.sif (S) mirror 1
- e4s-cuda80-aarch64-23.05.sif (S) mirror 1

Intel OneAPI (x86_64)
- ecpe4s/e4s-oneapi:23.05
- e4s-oneapi-x86_64-23.05.sif (S) mirror 1

https://e4s.io
Intel Compilers and MPI Libraries Now Accessible in E4S Containers:
A Breakthrough Collaboration Driving Productivity and Sustainability

• Background:
  – E4S provides a unified software stack of libraries and tools for portable performance on HPC systems, especially GPU-based systems.
  – E4S promises seamless portability for onsite and cloud-based workflows through its container-based approach.
  – Intel compilers and libraries available in E4S accelerates preparations for Aurora and future Intel-based GPU systems.
  – E4S eliminates the need for separate management of access to Intel compilers and libraries, benefiting users
  – Many important workflows, especially regression testing and turnkey usage for Intel platforms become feasible and easier

• The E4S-Intel agreement makes Intel compilers and MPI libraries available via E4S containers:
  – Enables full testing and execution of HPC libraries and tools on Intel platforms via E4S, including Aurora early access systems
  – Represents a win-win for DOE, Intel, and the broader E4S user community that is developing at other US agencies and industry

• The Intel agreement brings Intel in line with E4S builds that include AMD and NVIDIA tools.

• The E4S-Intel agreement is possible through the partnership of ECP and the E4S commercial provider, ParaTools, Inc.
E4S base container images allow users to customize their containers

- Intel oneAPI
- AMD ROCm
- NVIDIA NVHPC and CUDA

https://e4s.io
e4s-alc: a new tool to customize container images

Add to a base image:

- Spack packages
- OS packages
- Tarballs

https://github.com/E4S-Project/e4s-alc
e4s-alc: E4S à la carte – a tool to customize container images

Add packages to a container image:
• Spack packages
• OS packages (yum/apt/zypper)
• Add a tarball to a location
• Create a new container image
• Works for Docker & Singularity!

https://github.com/E4S-Project/e4s-alc
E4S 23.05 DOE LLVM and CI images

**DOE LLVM E4S Image**

This multi-architecture image contains E4S products compiled with DOE LLVM 16 and Flang using Spack

Multi-Arch (X86_64, PPC64LE, AARCH64)
- ece4s/e4s/doe llvm:23.05
- e4s-doelvm-x86_64-23.05.sif
- e4s-doelvm-ppc64le-23.05.sif

**Continuous Integration Images**

These are barebones operating system images which contain only essential build tools and python packages needed by Spack.

These images are intended to be used in continuous integration workflows where Spack is first cloned and then used to build and test software.

- **X86_64**
  - ece4s/ubuntu:22.04-runner-x86_64
  
- **PPC64LE**
  - ece4s/ubuntu:22.04-runner-ppc64le
  
- **AARCH64**
  - ece4s/ubuntu:22.04-runner-aarch64

https://e4s.io
E4S 23.05 Detailed Documentation for Bare-metal Installation

Extreme-scale Scientific Software Stack (E4S)
version 23.05

Exascale Computing Project (ECP) Software Technologies (ST) software, Extreme-scale Scientific Software Stack (E4S) v23.05, includes a subset of ECP ST software products, and demonstrates the target approach for future delivery of the full ECP ST software stack. Also available are a number of ECP ST software products that support a Spack package, but are not yet fully interoperable. As the primary purpose of the v23.05 is demonstrating the ST software stack release approach, not all ECP ST software products were targeted for this release. Software products were targeted primarily based on existing Spack package maturity, location within the scientific software stack, and ECP SDK developer experience with the software. Each release will include additional software products, with the ultimate goal of including all ECP ST software products.

E4S ReadTheDocs: Full Documentation.

E4S ReadTheDocs: Support Guide.

E4S Deployment Dashboard.

E4S v23.05 Release Notes PDF.

E4S v23.05 Spack Environment Notes.

E4S Manual Installation Instructions.

E4S Container Installation Instructions.

Recipes for building E4S images from scratch.

Prebuilt binaries used in E4S images are stored in the E4S Build Cache.
E4S 23.05 full featured container release on Dockerhub

docker pull ecpe4s/e4s-cuda:23.05

docker pull ecpe4s/e4s-cuda:latest

docker pull ecpe4s/e4s-cuda:23.05-cuda90

Architectures:
- x86_64
- aarch64
- ppc64le

Software:
- CUDA 12.0
- NVHPC 23.3
- oneAPI 2023.1
E4S 23.05 base container release on DockerHub

docker pull ecpe4s/e4s-base-cuda

Architectures:
- x86_64
- aarch64
- ppc64le

Software:
- CUDA 12.0
- NVHPC 23.3
- oneAPI 2023.1
Minimal Spack base image on Dockerhub

- Create custom container images
- 1M+ downloads!
23.05 Release: 100+ Official Products + dependencies (gcc, x86_64)

GPU runtimes
- AMD (ROCm)
  - 5.4.3
- NVIDIA (CUDA)
  - 12.0
- NVHPC
  - 23.3
- Intel oneAPI
  - 2023.1
23.05 Release: 100+ Official Products + dependencies (gcc, x86_64)
23.05 Release: 100 Official Products + dependencies (gcc, x86_64)

- py-libensemble
- py-parsl
- py-radical-saga
- qthreads
- quantum-espresso
- raja
- rempi
- scr
- slate
- sile
- silec
- stk
- strpack
- sundials
- superlu-dist
- swig
- sz
- tasmanian
- trilinos
- turbine
- umpire
- unifyf
tx
- upcxx
- variorum
- veloc
- visit
- vtk-m
- wanner90
- warpX
- xyce
- zfp

Languages:
- Julia with support for MPI, and CUDA
- Python

AI products with GPU support:
- Tensorflow
- Pytorch

EDA Tools:
- Xyce

3D Visualization:
- Paraview
- Visit
- TAU’s paraprop ...

E4S 23.05 adds support for NVIDIA A100 (sm80), V100 (sm70), and H100 (sm90) GPUs
E4S 23.05 Intel oneAPI 2023.1: Packages built with Intel compilers

Use of Intel oneAPI BaseKit and HPCToolkit is subject to acceptance of Intel EULA by the user
E4S 23.05 Intel oneAPI 2023.1: Packages built with Intel compilers

Use of Intel oneAPI BaseKit and HPCToolkit is subject to acceptance of Intel EULA by the user.

Singularity module avail

<table>
<thead>
<tr>
<th>advisor/latest</th>
<th>compiler32/latest</th>
<th>/opt/intel/oneapi/modulefiles</th>
</tr>
</thead>
<tbody>
<tr>
<td>advisor/2023.1.0</td>
<td>compiler32/2023.1.0</td>
<td>dnnl-cpu-tbb/latest</td>
</tr>
<tr>
<td>ccl/latest</td>
<td>dal/latest</td>
<td>inspector/latest</td>
</tr>
<tr>
<td>ccl/2021.9.0</td>
<td>dal/2023.1.0</td>
<td>dnnl-cpu-tbb/2023.1.0</td>
</tr>
<tr>
<td>click/latest</td>
<td>debugger/latest</td>
<td>dnnl/latest</td>
</tr>
<tr>
<td>click/2021.7.3</td>
<td>debugger/2023.1.0</td>
<td>dnnl2023.1.0</td>
</tr>
<tr>
<td>compiler-rt/latest</td>
<td>dev-utilties/latest</td>
<td>dpl/latest</td>
</tr>
<tr>
<td>compiler-rt/2023.1.0</td>
<td>dev-utilties/2023.1.0</td>
<td>dpl/2022.1.0</td>
</tr>
<tr>
<td>compiler-rt/2023.1.0</td>
<td>dnnl-cpu-gomp/latest</td>
<td>icc/latest</td>
</tr>
<tr>
<td>compiler-rt/2023.1.0</td>
<td>dnnl-cpu-gomp/2023.1.0</td>
<td>icc/2023.1.0</td>
</tr>
<tr>
<td>compiler/latest</td>
<td>dnnl-cpu-omp/latest</td>
<td>icc2023.1.0</td>
</tr>
<tr>
<td>compiler/2023.1.0</td>
<td>dnnl-cpu-omp/2023.1.0</td>
<td>init_openmpi/2023.1.0</td>
</tr>
<tr>
<td>compiler/2023.1.0</td>
<td>dnnl-cpu-omp/2023.1.0</td>
<td>init_openmpi/2023.1.0</td>
</tr>
</tbody>
</table>

---

Use of Intel oneAPI BaseKit and HPCToolkit is subject to acceptance of Intel EULA by the user.
E4S Support for ROCm variants for MI250X (gfx90a) on x86_64

Singularity> spack find --linux-ubuntu20.04-x86_64 / gcc@11.1.0

adios@1.13.1
adios2@2.9.0
alquimia@1.0.10
aml@0.2.0
amrex@23.05
amrex@23.05
arborx@1.3
arborx@1.3
archer@0.2.0
argobots@0.1
ascent@0.9.1
axiom@0.7.0
bolt@2.0
boost@1.79.0
bricks@0.6
butterflypack@2.2.2
cabana@0.5.0
cabana@0.5.0
cabana@0.5.0
caliper@0.9.0
caliper@0.9.0
chait@2022.03.0
chait@2022.03.0
chait@2022.03.0

== 153 installed packages

e4s 23.05 supports AMD MI100 (gfx908) as well as MI250X (gfx90a) GPUs
E4S 23.05 supports AMD MI100 (gfx908) as well as MI250X (gfx90a) GPUs
E4S 23.05 DOE LLVM Release: x86_64, ppc64le, and aarch64

Singularity> spack find -x
-- linux-ubuntu20.04-x86_64 / clang016.0.2 ------------------------
adios@1.13.1 cabana@0.5.0 globalarrays@0.8.2 heffte@2.3.0
ami@0.2.0 chai@2022.03.0 gmp@0.2.1 hypre@2.28.0
amrex@23.05 charliecloud@0.32 gotcha@1.8.4 legion@23.03.0
arborx@1.3 flit@2.1.0 h5bench@1.3 libnm@0.1.0
argobots@0.1.1 flux-core@0.49.0 hdf5-vol-async@1.5 libquok@0.3.1
bolt@2.0 gasnet@2023.3.0 hdf5-vol-log@1.4.0 libunwind@0.6.2
mfem@0.4.5.2 parsec@3.2209 sundials@0.6.5.1 umpire@2022.03.1
mpark-variant@1.4.0 ptd@0.25.1 superlu@0.3.0 upcxx@2023.3.0
mpich@0.4.1.1 plumed@0.2.8 swig@4.0.2-fortran
ncmp@1.9.0.1 pumi@2.2.7 tasmanian@0.9.7
nco@0.5.1 qthreads@1.16 turbine@0.3.0
papyrus@0.1.0 stc@0.9.0 umap@2.1.0

-- linux-ubuntu20.04-x86_64 / gcc@11.1.0 -------------------------
cmake@3.26.3 llvm-doe@16.0.2
Singularity> spack find -x
-- linux-ubuntu20.04-ppc64le / clang@16.0.2 ----------------------
adios@1.13.1 cabana@0.5.0 globalarrays@0.8.2 heffte@2.3.0
ami@0.2.0 chai@2022.03.0 gmp@0.2.1 hypre@2.28.0
amrex@23.05 charliecloud@0.32 gotcha@1.8.4 legion@23.03.0
arborx@1.3 flit@2.1.0 h5bench@1.3 libnm@0.1.0
argobots@0.1.1 flux-core@0.49.0 hdf5-vol-async@1.5 libquok@0.3.1
bolt@2.0 gasnet@2023.3.0 hdf5-vol-log@1.4.0 libunwind@0.6.2
mfem@0.4.5.2 parsec@3.2209 sundials@0.6.5.1 umpire@2022.03.1
mpark-variant@1.4.0 ptd@0.25.1 superlu@0.3.0 upcxx@2023.3.0
mpich@0.4.1.1 plumed@0.2.8 swig@4.0.2-fortran
ncmp@1.9.0.1 pumi@2.2.7 tasmanian@0.9.7
nco@0.5.1 qthreads@1.16 turbine@0.3.0
papyrus@0.1.0 stc@0.9.0 umap@2.1.0

-- linux-ubuntu20.04-ppc64le / gcc@11.1.0 -------------------------
cmake@3.26.3 llvm-doe@16.0.2
Singularity> spack find -x
-- linux-ubuntu20.04-aarch64 / clang@16.0.2 ----------------------
adios@1.13.1 cabana@0.5.0 globalarrays@0.8.2 heffte@2.3.0
ami@0.2.0 chai@2022.03.0 gmp@0.2.1 hypre@2.28.0
amrex@23.05 charliecloud@0.32 gotcha@1.8.4 legion@23.03.0
arborx@1.3 flit@2.1.0 h5bench@1.3 libnm@0.1.0
argobots@0.1.1 flux-core@0.49.0 hdf5-vol-async@1.5 libquok@0.3.1
bolt@2.0 gasnet@2023.3.0 hdf5-vol-log@1.4.0 libunwind@0.6.2
mfem@0.4.5.2 parsec@3.2209 sundials@0.6.5.1 umpire@2022.03.1
mpark-variant@1.4.0 ptd@0.25.1 superlu@0.3.0 upcxx@2023.3.0
mpich@0.4.1.1 plumed@0.2.8 swig@4.0.2-fortran
ncmp@1.9.0.1 pumi@2.2.7 tasmanian@0.9.7
nco@0.5.1 qthreads@1.16 turbine@0.3.0
papyrus@0.1.0 stc@0.9.0 umap@2.1.0

-- linux-ubuntu20.04-aarch64 / gcc@11.1.0 -------------------------
cmake@3.26.3 llvm-doe@16.0.2
E4S Build Cache for Spack 0.19.1 hosted at U. Oregon

- Over 100K binaries!
- No need to recompile from source code.
e4s-cl: A tool to simplify the launch of MPI jobs in E4S containers

- E4S containers support replacement of MPI libraries using MPICH ABI compatibility layer and Wi4MPI [CEA] for OpenMPI replacement.
- Applications binaries built using E4S can be launched with Singularity using MPI library substitution for efficient inter-node communications.
- e4s-cl is a new tool that simplifies the launch and MPI replacement.
  - e4s-cl init --backend [singularity|shifter|docker] --image <file> --source <startup_cmds.sh>
  - e4s-cl mpirun -np <N> <command>
- Usage:
  ```bash
e4s-cl init --backend singularity --image ~/images/e4s-gpu-x86.sif --source ~/source.sh
cat ~/source.sh
  . /spack/share/spack/setup-env.sh
  spack load trilinos+cuda cuda_arch=80
e4s-cl mpirun -np 4 .a.out
```

https://github.com/E4S-Project/e4s-cl
E4S: A Container Platform for High Throughput Computing

- E4S singularity containers supporting NVIDIA GPUs and Intel runtimes are now available on OSG-HTC [https://osg-htc.org]:
  - `/cvmfs/singularity.opensciencegrid.org/ecpe4s/e4s-cuda:23.05`
  - `/cvmfs/singularity.opensciencegrid.org/ecpe4s/e4s-oneapi:23.05`

- E4S containers include vendor runtimes (NVIDIA CUDA/NVHPC and Intel oneAPI)

- E4S containers include full-featured Python with support for AI/ML tools with support for GPUs
  - PyTorch 2.0.0, TensorFlow 2.12.0
  - Numpy, scipy, keras, matplotlib,…
  - `pip3 install --user` other packages in your workflow

- E4S containers include 100+ HPC tools installed using the Spack package manager

- E4S features base containers as well as full-featured containers and tools to customize containers

- E4S containers will continue to be supported in future releases on OSG-HTC and provides a viable platform for tool integration for high throughput computing
This research was supported by the Exascale Computing Project (17-SC-20-SC), a joint project of the U.S. Department of Energy’s Office of Science and National Nuclear Security Administration, responsible for delivering a capable exascale ecosystem, including software, applications, and hardware technology, to support the nation’s exascale computing imperative.

Thank you to all collaborators in the ECP and broader computational science communities. The work discussed in this presentation represents creative contributions of many people who are passionately working toward next-generation computational science.

https://www.exascaleproject.org