SC18 Dallas, TX | hpc inspires.

OpenMP 4.5 Validation and Verification Suite for Device offload

Sunita Chandrasekaran **Jose Monsalve Diaz**

Swaroop Pophale Oscar Hernandez David E. Bernholdt

University of Delaware

ECP EXASCALE COMPUTING PROJECT

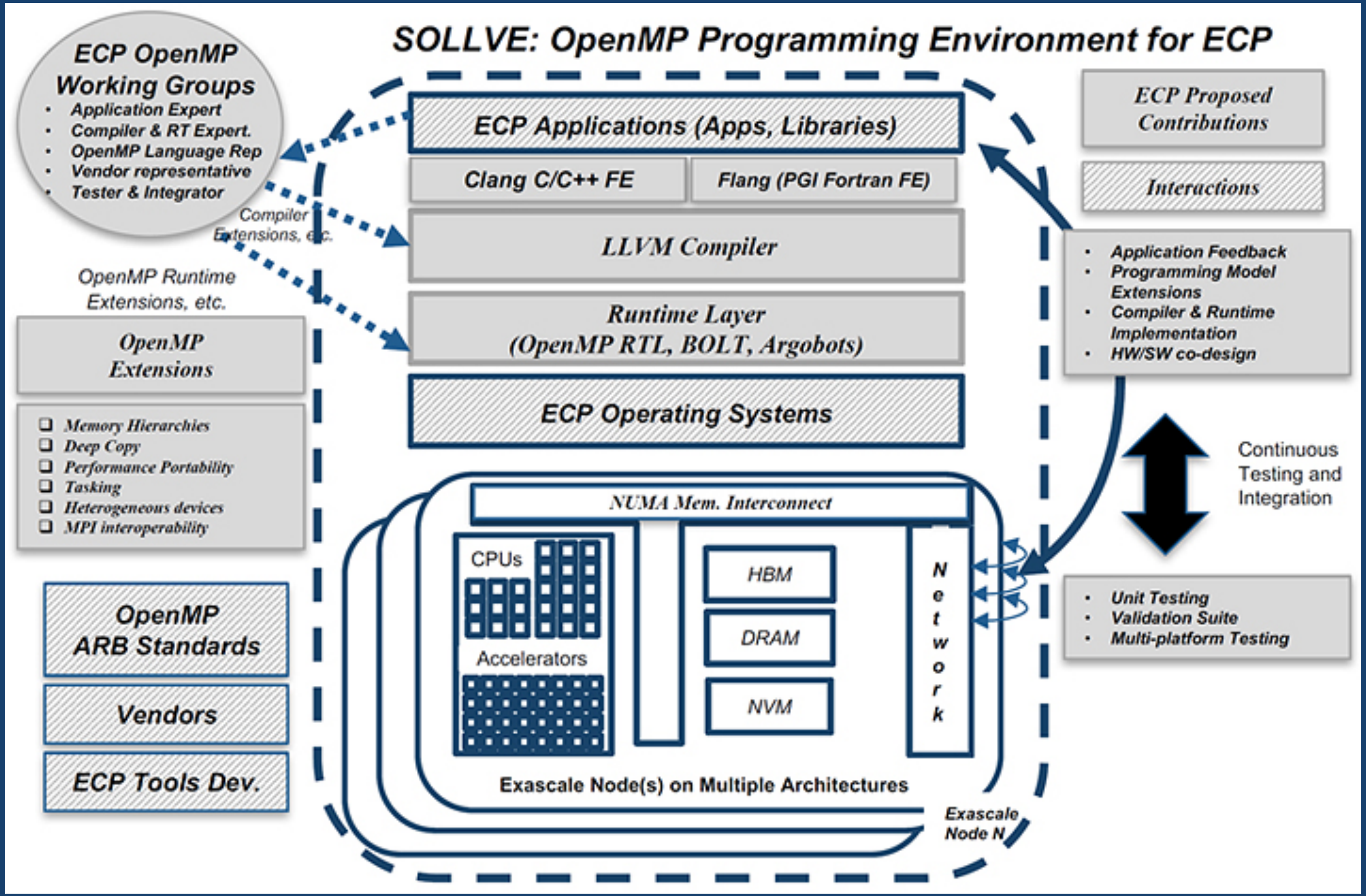
Oak Ridge National Laboratory

Outline

- Introduction
  - Problem Statement and Contributions
  - Related work
- OpenMP 4.5 offloading
- Methodology
  - Test design
  - Infrastructure design
  - Results logs and reports
- Results

Introduction

The ECP SOLLVE Project



SOLLVE: OpenMP Programming Environment for ECP

ECP OpenMP Working Groups

- Application Expert
- Compiler & RT Expert.
- OpenMP Language Rep
- Vendor representative
- Tester & Integrator

Compiler Extensions, etc.

OpenMP Runtime Extensions, etc.

OpenMP Extensions

- ❑ Memory Hierarchies
- ❑ Deep Copy
- ❑ Performance Portability
- ❑ Tasking
- ❑ Heterogeneous devices
- ❑ MPI interoperability

OpenMP ARB Standards

Vendors

ECP Tools Dev.

ECP Applications (Apps, Libraries)

Clang C/C++ FE

Flang (PGI Fortran FE)

LLVM Compiler

Runtime Layer (OpenMP RTL, BOLT, Argobots)

ECP Operating Systems

NUMA Mem. Interconnect

CPUs

Accelerators

HBM

DRAM

NVM

Network

Exascale Node(s) on Multiple Architectures

Exascale Node N

ECP Proposed Contributions

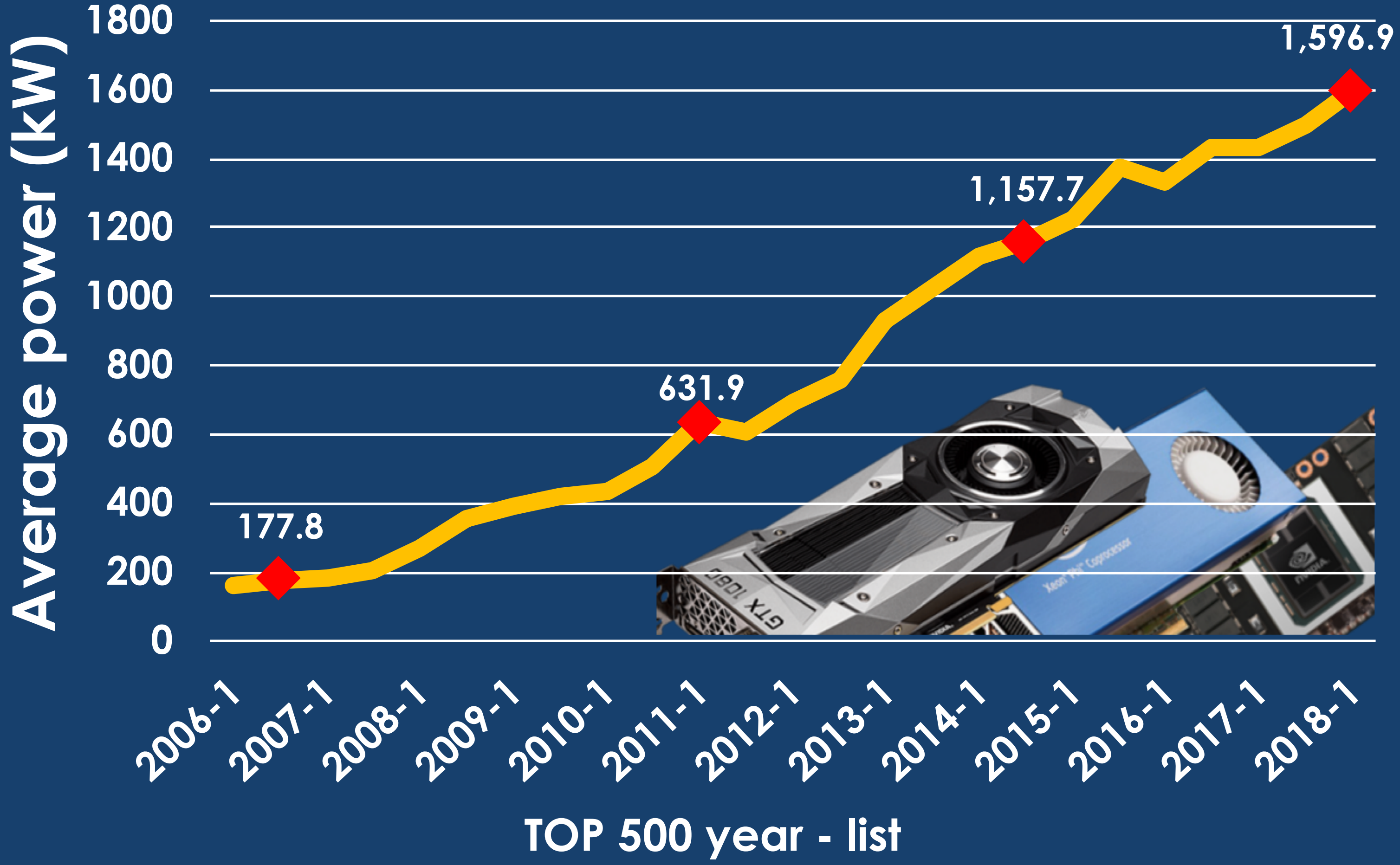
Interactions

- Application Feedback
- Programming Model Extensions
- Compiler & Runtime Implementation
- HW/SW co-design

Continuous Testing and Integration

- Unit Testing
- Validation Suite
- Multi-platform Testing

Why Accelerator devices?



- Power wall limited single core performance and big cores
- Simpler cores and considerably larger core count
- But there are still programmability challenges...

Problem Statement

As the OpenMP specification continues to grows, **what are the methodologies and tools** to measure the level of compliance of a compiler implementation and supported system with respect to the OpenMP specification document?

Motivation

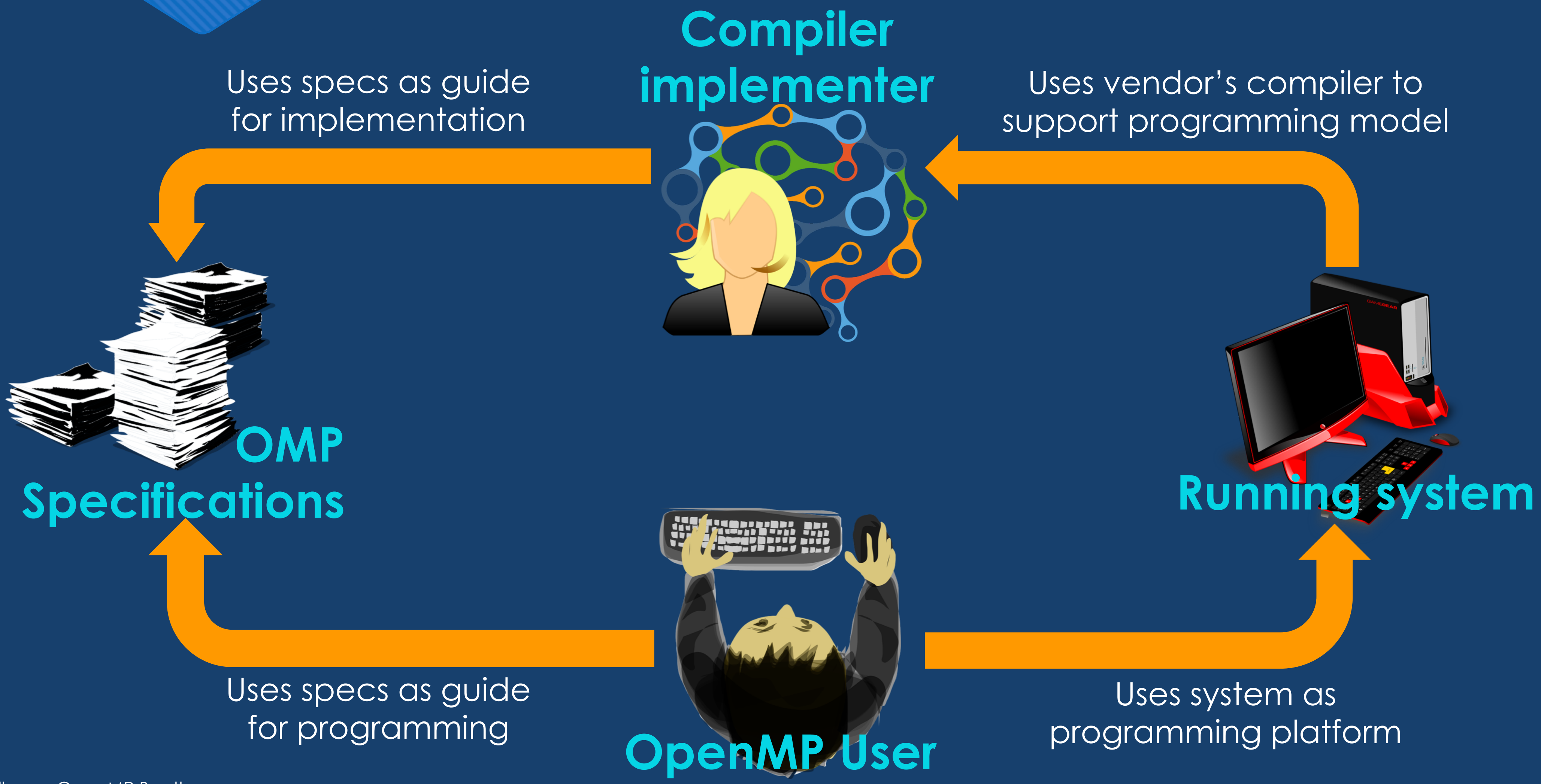
Programming model specifications is a

“Legal” document that binds **compiler implementation** and **users**

- Compiler developers use the specification to define the compiler behavior. To claim support the specs must be respected.
- Users do not need to learn all the implementation specific aspects to use the programming model as long as they know the specs

What about the system where the user is running on?

Motivation



Compiler implementer

Uses specs as guide for implementation

Uses vendor's compiler to support programming model

OMP Specifications

Running system

Uses specs as guide for programming

OpenMP User

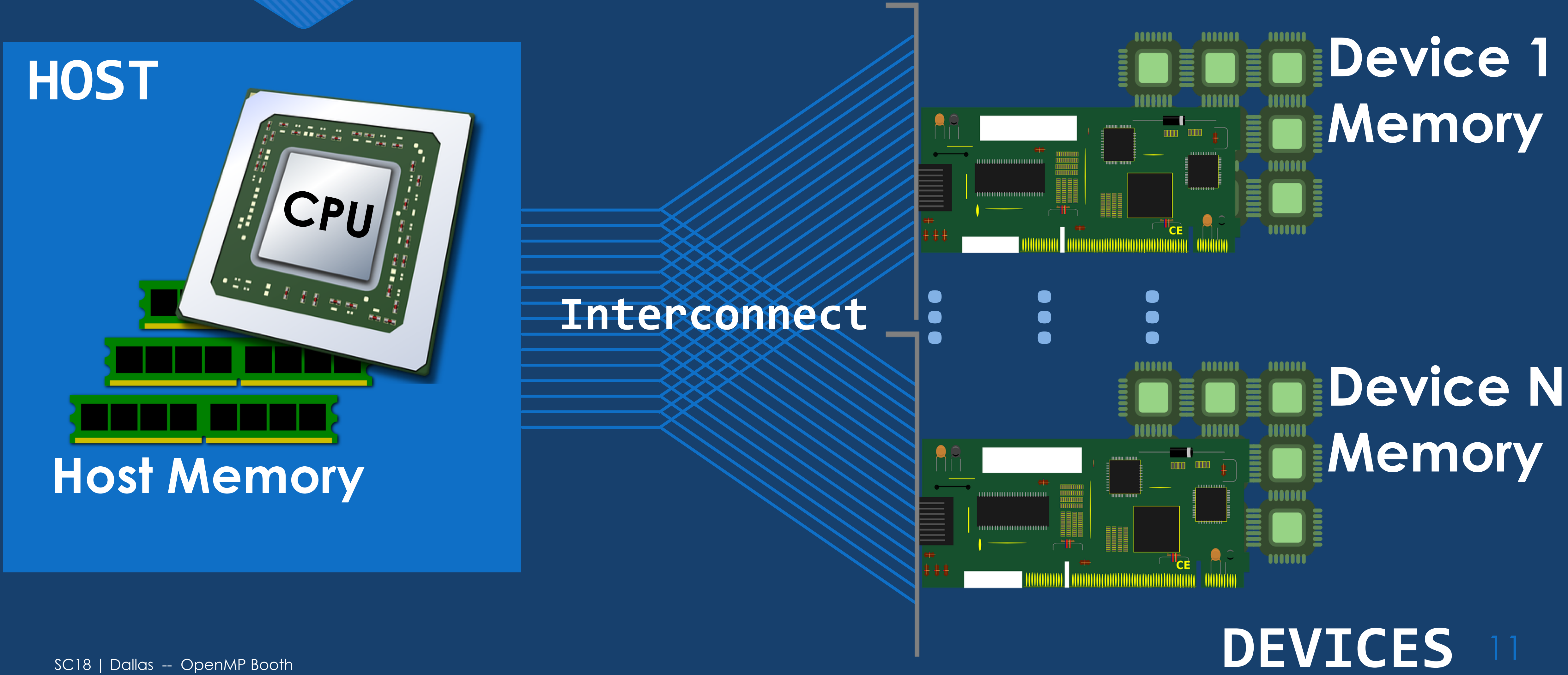
Uses system as programming platform

Why not ____ ?

- **Examples as tests?**
  - Lack of coverage
  - Different purpose
- **Automatic testing?**
  - Language interpretation of the Specs (Human required)
- **Use vendor tests?**
  - Vendor specific methodology
  - Biased interpretation

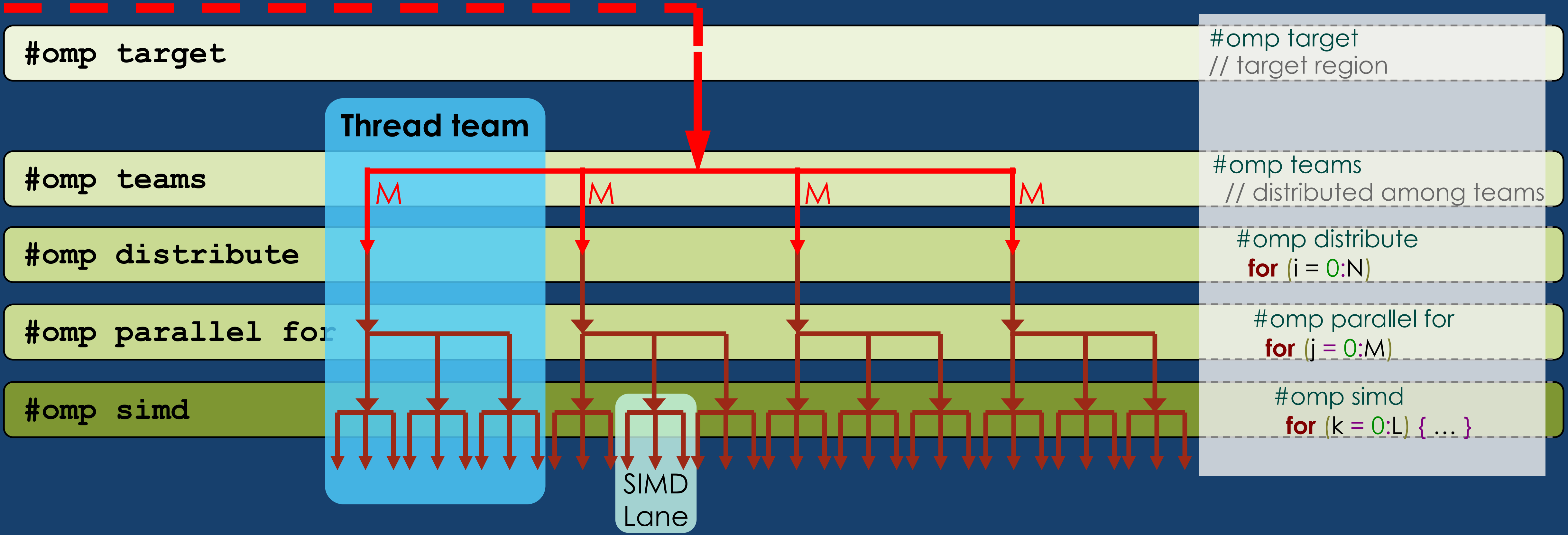
OpenMP 4.5 Offloading model

OpenMP 4.5 Machine model



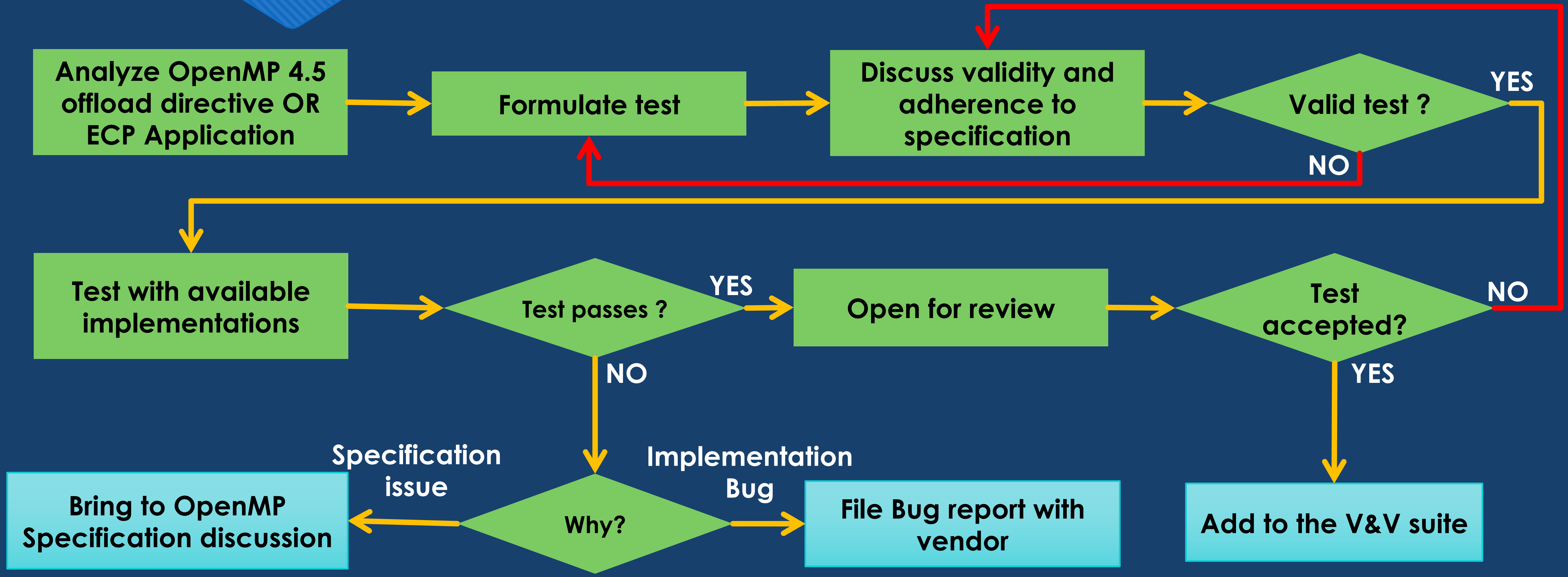
OpenMP 4.5 Execution Model

HOST to TARGET OFFLOADING



Tests suite methodology

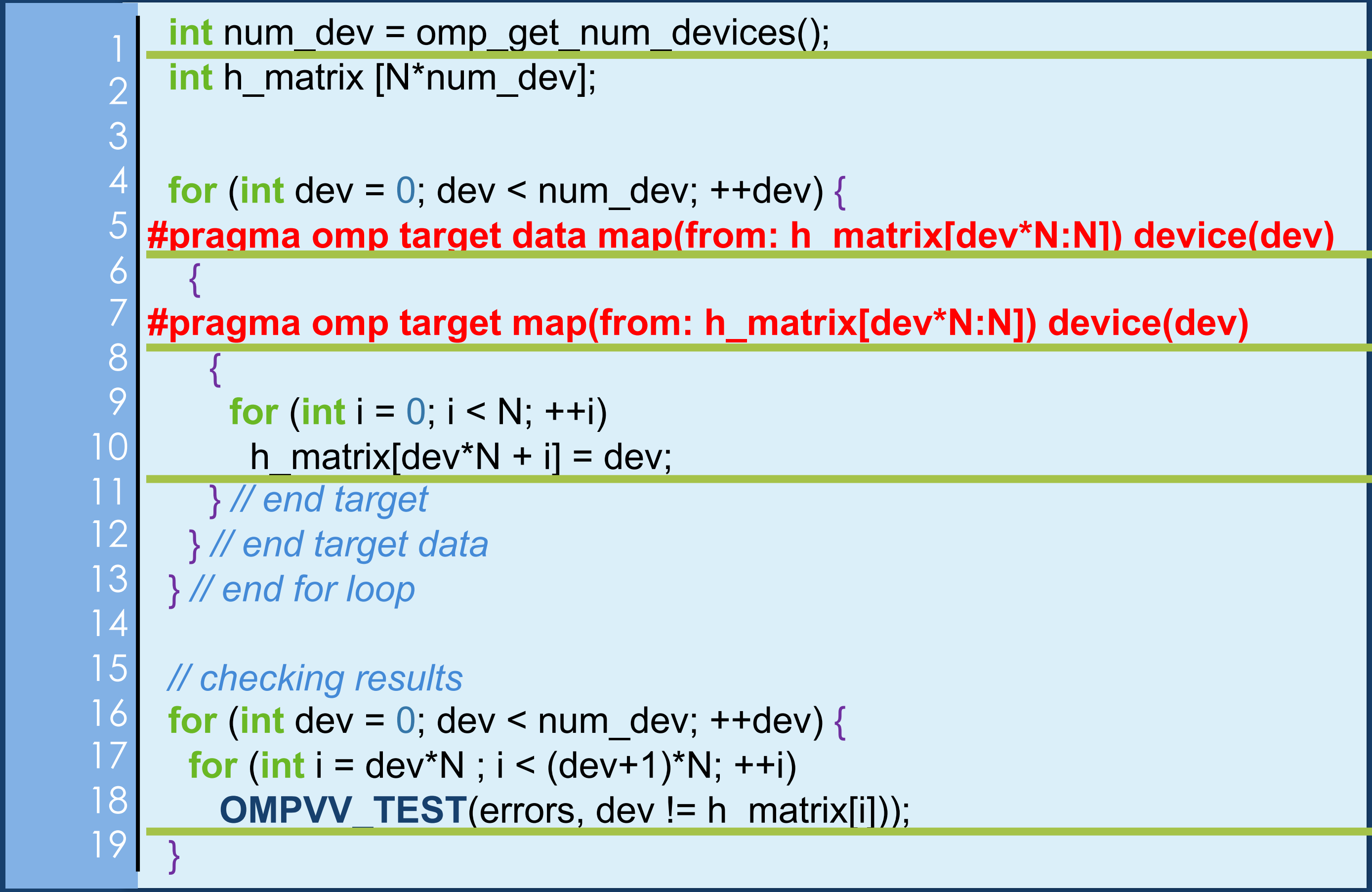
Methodology



Analyze OpenMP 4.5 offload directive OR ECP Application
Formulate test
Discuss validity and adherence to specification
Valid test ?
YES
NO
Test with available implementations
Test passes ?
YES
NO
Open for review
Test accepted?
NO
YES
Specification issue
Why?
Implementation Bug
Bring to OpenMP Specification discussion
File Bug report with vendor
Add to the V&V suite

Test design

Data mapping on multiple devices



```
int num_dev = omp_get_num_devices();
int h_matrix [N*num_dev];

for (int dev = 0; dev < num_dev; ++dev) {
#pragma omp target data map(from: h_matrix[dev*N:N]) device(dev)
  {
#pragma omp target map(from: h_matrix[dev*N:N]) device(dev)
    {
      for (int i = 0; i < N; ++i)
        h_matrix[dev*N + i] = dev;
    } // end target
  } // end target data
} // end for loop

// checking results
for (int dev = 0; dev < num_dev; ++dev) {
  for (int i = dev*N ; i < (dev+1)*N; ++i)
    OMPVV_TEST(errors, dev != h_matrix[i]));
}
```

Get number of available devices

Map data from selected device to host

Create target region on selected device

Computation offloaded to selected device

Compare results with expected

Test design

Target teams distribute firstprivate (segment)

```
int a[N] = 1; b[N] = x; c[N] = 2*x; d[N] = 0;
Int privatized = 1;
#pragma omp target teams distribute firstprivate(privatized) \
map(from: d[:]) map(to: a[:], b[:], c[:])
for (int x = 0; x < N; ++x) {
  for (int y = 0; y < a[x] + b[x]; ++y) {
    privatized++;
  }
  d[x] = c[x] * privatized; privatized = 0;
}

for (int x = 0; x < N; ++x){
    OMPVV_TEST_AND_SET_VERBOSE(errors, d[x] != (2 + x) * 2 * x);
}
```

Variable to privatize

Privatize and initialize value

Possible data race if privatization fails

d[] serves as probe to check the value

Check the expected value of d[]

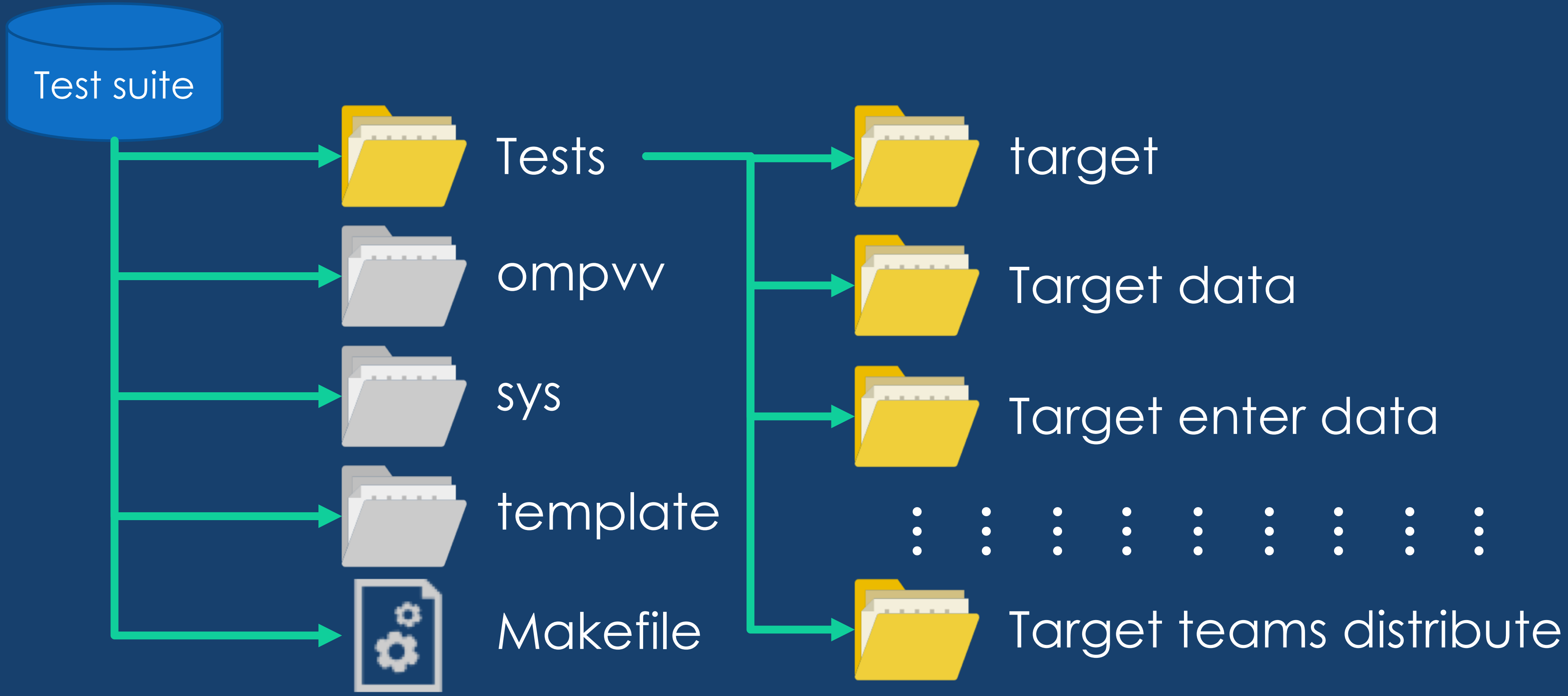
Infrastructure design

Our infrastructure is based on a **Makefile + scripts**

Design parameters:

- Portability across multiple **compilers** and **systems** and easy to use
  - Support for different compiler options
  - Support for Lua-like Modules and batch schedulers (Usually used in HPC clusters)
- Fast test addition and modification
- Divided compilation and execution phases
- Subset of tests selection for partial execution

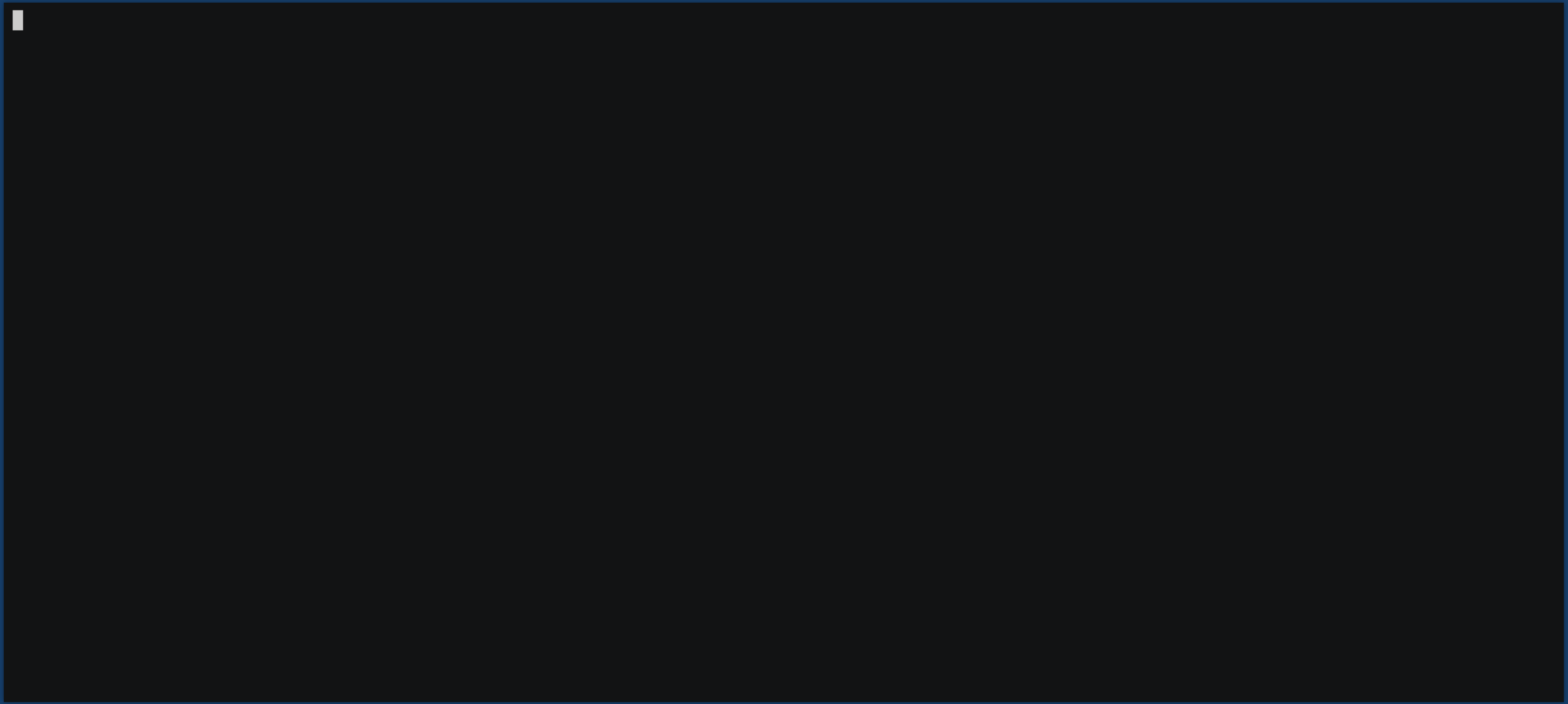
Folder structure



Makefile Options

Option	Description
VERBOSE=1	Make output verbosity
VERBOSE_TESTS=1	Test output verbosity
LOG=1	Enable test output logging
LOG_ALL=1	Enables Make output and test output logging
SYSTEM=sys_name	Include system definition file
ADD_BATCH_SCHED=1	Add batch schedule line before running a test
MODULE_LOAD=1	Add *module load* commands before every command
SOURCES=file.(c,cpp,F90)	Select a subset of tests

Makefile Simple Example



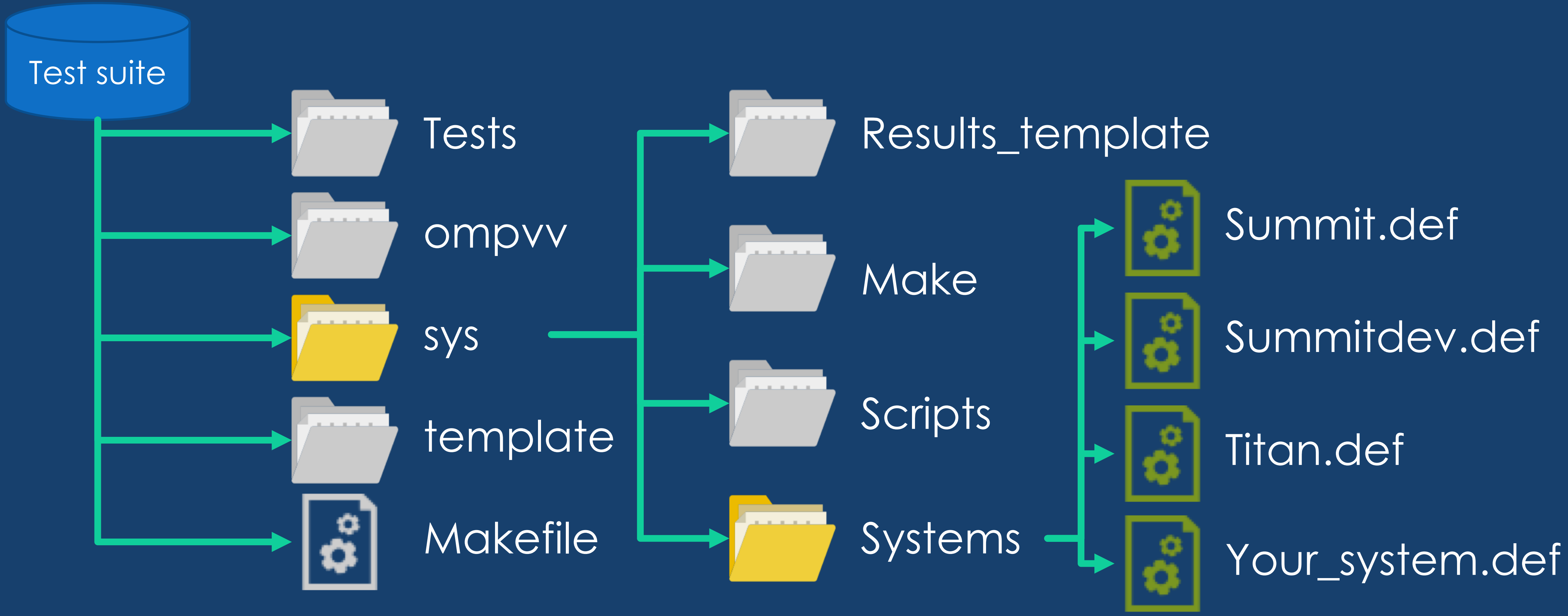
Makefile Rules

Option	Description
all	Compile and run
compile	Compile only
run	Run previously compiled tests
clean	Clean previously compiled tests
compilers	List available compilers (SYSTEM option dependent)
report_json	Parse raw logs and produce results.json file
report_html	Use JSON file and template to produce html visualization of results

System support

- Specify system specific support for:
  - Linux-line Module loading for each compiler
  - Batch scheduling execution lines (e.g. jsrun and aprun)
  - Get compiler version (command)
  - Specify compiler specific flags (CFLAGS, CXXFLAGS)

Systems folder



Results
Log files

- 3 Different Log formats:
  - RAW Format:
    - Format specific output log.
    - Single file per test,
    - Single section per action delimited by header and footer with system and runtime information
  - JSON FILE:
    - Translation process from parsing the raw format
  - HTML (visual):
    - Template that uses the JSON file to load results into a visualization table

Experimental setup

System	Model	Processors	Cores/node	Threads/node	Memory	Accelerator	Complers
Titan	Cray XK7	AMD Opteron 6274	16	16	32 GB	1 NVIDIA K20X	CCE 8.7.2
Summitdev	IBM S822LC	2x Power8	20	160	256 GB	4 NVIDIA P100	GCC 7.1.1 Clang 3.8.0 XLC 13.1.6
Summit	IBM AC922	2x Power9	42	168	512 GB	6 NVIDIA V100	Clang 3.8.0 XLC 13.1.7

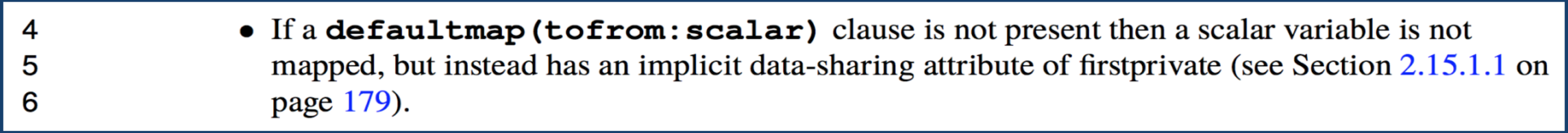
Current results snapshots

Visit our website for more

	Summitdev					Summit **		Titan
OpenMP Features	GCC 7.1.1	gfortran 7.1.1	Clang CORAL 3.8.0	XLC 13.1.6	XLF 15.1.7	Clang CORAL 3.8.0	XLC 13.1.7	CCE 8.7.2
target	14/15	13/13	15/15	14/15	12/14	12/14	11/14	13/14
target data	5/5	10/10	6/6	6/6	8/10	6/6	6/6	3/6
target enter/exit data	6/6	4/4	6/6	6/6	5/5	6/7	6/7	5/7
target enter data	7/7	6/6	7/7	7/7	7/7	6/7	6/7	5/7
target update	5/5	-	5/5	5/5		5/5	4/5	4/5
target teams distribute **	12/18	2/8	10/18	12/18	2/8	-	-	9/11
target teams distribute parallel for **	13/14	-	13/14	11/14	-	-	-	10/14

Example Implementation discrepancy

Specifications (Section 2.15.5 - page 216):



- If a **`defaultmap(tofrom:scalar)`** clause is not present then a scalar variable is not mapped, but instead has an implicit data-sharing attribute of firstprivate (see Section 2.15.1.1 on page 179).

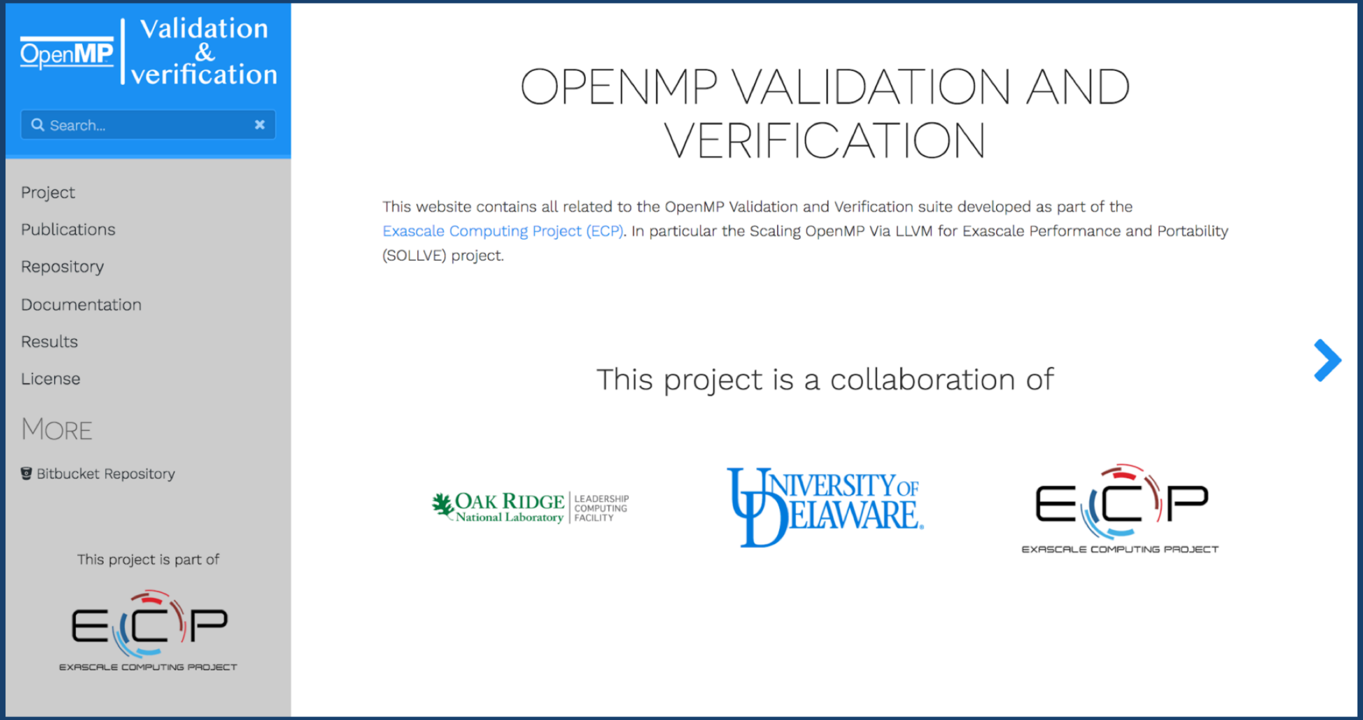
Test:

```
enum { VAL1 = 1, VAL2, VAL3, VAL4} scalar_enum = VAL1
#pragma omp target
  {
    scalar_enum = VAL4;
  }
OMPVV_TEST_AND_SET_VERBOSE(errors, scalar_enum != VAL1);
```

RESOLVED

Failed condition

Visit our website
https://crpl.cis.udel.edu/ompvvsollve/


OpenMP
Validation & verification
Search...
Project
Publications
Repository
Documentation
Results
License
MORE
Bitbucket Repository
This project is part of
ECP EXASCALE COMPUTING PROJECT

OPENMP VALIDATION AND VERIFICATION

This website contains all related to the OpenMP Validation and Verification suite developed as part of the Exascale Computing Project (ECP). In particular the Scaling OpenMP Via LLVM for Exascale Performance and Portability (SOLLVE) project.

This project is a collaboration of

OAK RIDGE National Laboratory | LEADERSHIP COMPUTING FACILITY
UNIVERSITY OF DELAWARE
ECP EXASCALE COMPUTING PROJECT

Contact information:

Jose Monsalve (josem@udel.edu)

Swaroop Pophale (pophaless@ornl.gov)

Kyle Friedline (utimatu@udel.edu)

Oscar Hernandez (oscar@ornl.gov)

Sunita Chandrasekaran (schandra@udel.edu)

Work supported by the **U.S. Department of Energy**, Office of Science, the **Exascale Computing Project (17-SC-20-SC)**, a collaborative effort of the **U.S. Department of Energy Office of Science** and the **National Nuclear Security Administration** under contract number **DE-AC05-00OR22725.**

We also thank all of those who directly or indirectly have help this project.

Copyright

Gavel taken from(Chris Potter - Modifier: Ibrahim.ID):
https://commons.wikimedia.org/wiki/File:3D_png_Judges_Gavel.png

Logos under CC0:

- https://pixabay.com/en/papers-stack-heap-documents-576385/
- https://pixabay.com/en/gear-wheel-gearwheel-gear-cogs-310906/
- https://pixabay.com/en/user-top-view-office-keyboard-154199/
- https://pixabay.com/en/brain-cognition-design-art-2029391/
- https://pixabay.com/en/computer-workstation-server-monitor-158743/
- https://pixabay.com/en/cpu-processor-intel-amd-chip-152656/
- https://pixabay.com/en/microprocessor-processor-cpu-chip-152599/
- https://pixabay.com/en/computer-cyber-circuitry-circuits-3163436/
- https://pixabay.com/en/ram-computer-memory-module-148579/