INCITE GPU HACKATHON
MAY 7, 2025

OCCA Portability Framework

KRIS ROWE
Assistant Computational Scientist
Performance Engineering Group

U.S. DEPARTMENT OF ENERGY
Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

Argonne NATIONAL LABORATORY
Argonne Leadership Computing Facility

What is OCCA?

https://github.com/libocca/occa

Host API

- Unified models for device, memory, etc.
- Runtime selection of backend
- JIT compilation and caching of kernels

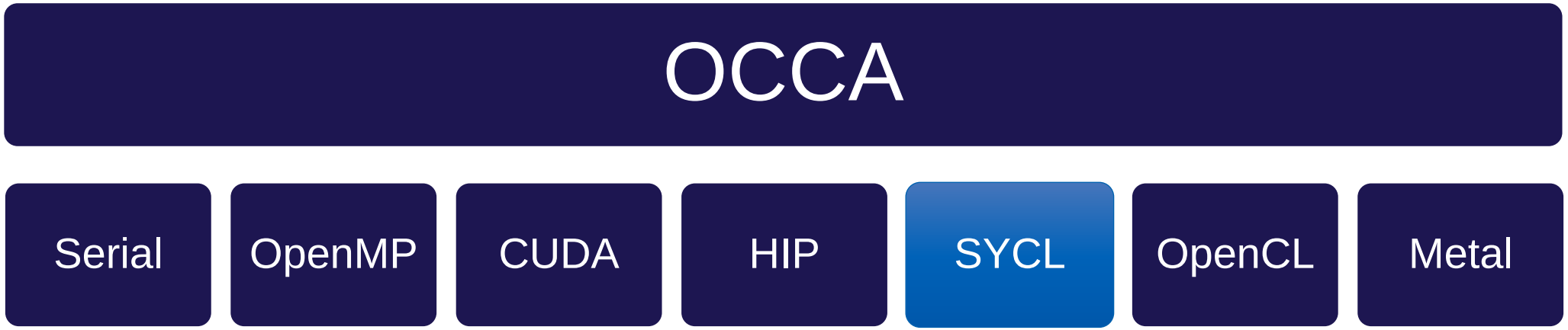
Kernel Language

- Directive based extension to C language
- Transparent translation to backend code

Command Line Tool

- Hardware information
- Available modes (backends)
- Environment variables
- Offline kernel translation and compilation

OCCA Backends



Example Program

```cpp
occa::device selectDevice(int device_id=0, int platform_id=0) {
  occa::json device_properties;
  device_properties["device_id"] = device_id;
  device_properties["platform_id"] = platform_id;
  device_properties["mode"] = "Serial"; // Default mode

  std::vector<std::string> preferred_modes = {"CUDA","HIP","dpcpp"};
  for (auto& mode : preferred_modes) {
    if (occa::modeIsEnabled(mode)) {
      device_properties["mode"] = mode;
      break;
    }
  }
  return occa::device(device_properties);
}
```

```cpp
#include <occa.hpp>

namespace {···

int main() {

  occa::device occa_device = selectDevice();

  const int N = 1024 * 1024 * 1024;
  const double alpha = 1.0;
  std::vector<double> hX(N,1.0);
  std::vector<double> hY(N,1.0);

  // Allocate + initialize device memory
  occa::memory dX = occa_device.malloc<double>(hX.size(),hX.data());
  occa::memory dY = occa_device.malloc<double>(hY.size(),hY.data());

  // Create a kernel defined in an external file
  const std::string kernel_name = "daxpy";
  const std::string kernel_file = "daxpy.okl";
  occa::kernel daxpy_kernel = occa_device.buildKernel(kernel_file,kernel_name);
  // Kernel is jitted during construction

  // Call the kernel like any function
  daxpy_kernel(N,dX,dY);
  // Work on device occurs in-order

  // Copy back to the host
  dY.copyTo(hY.data());
  // Verify the results ...

  // OCCA frees memory automatically
  return 0;
}
```

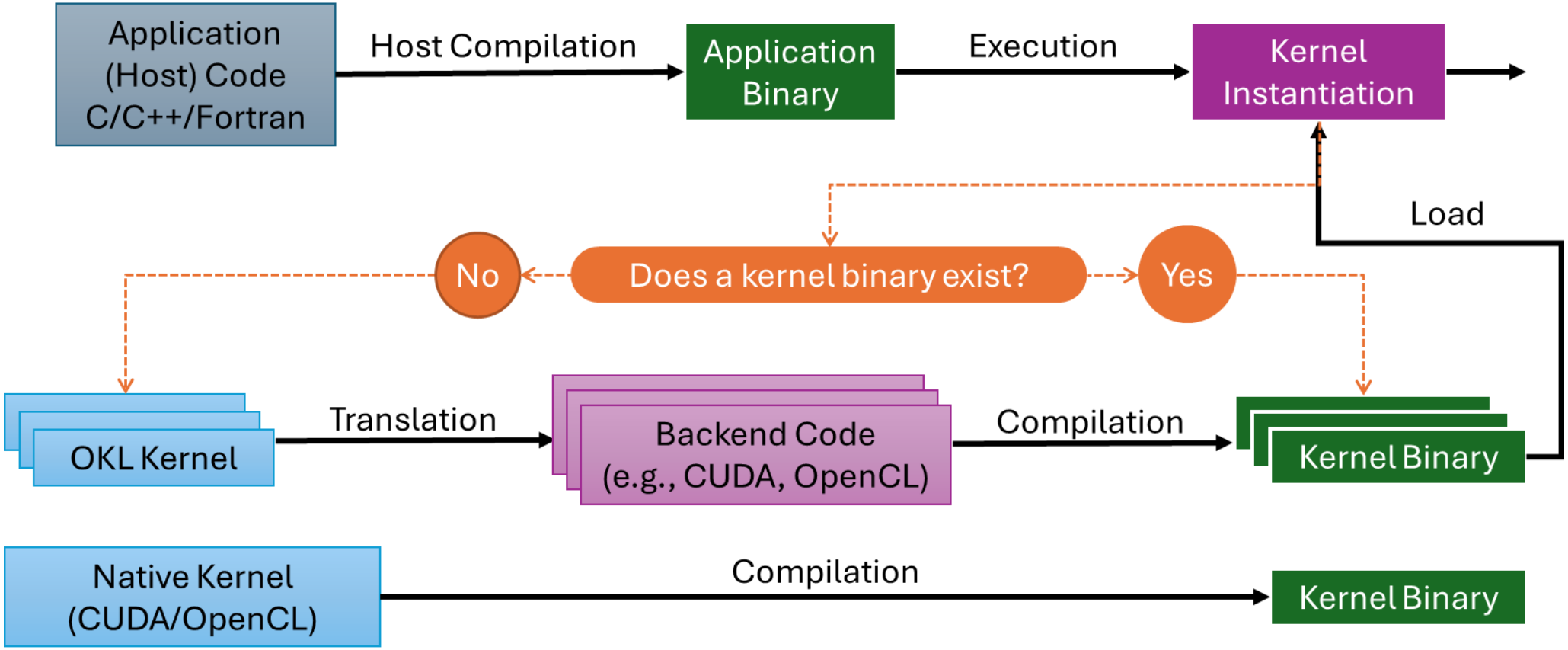
Mapping of Core OCCA Abstractions

OCCA	CUDA	HIP	SYCL	OpenCL
Device	CUdevice	hipDevice_t	sycl::device	cl_device_id
Memory	CUdeviceptr	hipDeviceptr_t	SYCL USM	cl_mem
Kernel	CUfunction	hipFunction_t	nd-range parallel_for	cl_kernel
Stream	CUstream	hipStream_t	sycl:queue (in-order)	cl_command_queue
StreamTag	CUevent	hipEvent_t	sycl::event	cl_event

Kernels

- Kernels can be provided to OCCA
  - In a separate source file
  - Using strings
- Kernel source language can be
  - Backend specific (e.g., CUDA, OpenCL)
  - OCCA Kernel Language (OKL)
- Jitting and caching of kernels looks similar for all the above

Kernel Jitting



OCCA Kernel Language

- Directive based extension to C
- Enables kernel portability
- Translated to backend-specific code

```
@kernel void axpy(int N, double alpha, const double* x, double* y) {
  @outer for (int j = 0; j < N; j += BLOCK_SIZE) {
    @inner for (int i = 0; i < BLOCK_SIZE; ++i) {
      const int n = i + j;
      if (n < N) y[n] += alpha * x[n];
    }
  }
}
```

Mapping of OKL Attributes

OKL	CUDA / HIP	SYCL	OpenCL
@kernel	__global__	extern “C” void	__kernel__
@outer	Thread Block ID	Work-group ID	Work-group ID
@inner	Thread ID	Work-Item ID (local)	Work-Item ID (local)
@shared	__shared__	group_local_memory	__local__
@barrier	__syncthreads	group_barrier	work_barrier

Command-line Tool

- OCCA_ROOT/bin/occa
- Display system info
- Translate OKL source
- Print OCCA env variables

```
========+======================+==============================
 dpcpp  | Platform 0           | Intel(R) Level-Zero
        |----------------------+------------------------------
        | Device 0             | Intel(R) Graphics [0x020a]
        | Device Type          | gpu
        | Compute Cores        | 960
        | Global Memory        | 25.47 GB
        | Local Memory         | 64 KB
        |----------------------+------------------------------
        | Platform 1           | Intel(R) OpenCL HD Graphics
        |----------------------+------------------------------
        | Device 0             | Intel(R) Graphics [0x020a]
        | Device Type          | gpu
        | Compute Cores        | 960
        | Global Memory        | 25.47 GB
        | Local Memory         | 64 KB
========+======================+==============================
```

nekRS x OCCA

- Use convenient wrappers provided to allocate memory from device memory pool
- Ensure you are using SYCL AOT flags for kernel jitting
- See nekRS Aurora job script for details

Getting Support

- Ask a question on GitHub Discussions (best!)
  - https://github.com/libocca/occa
- OCCA Slack for complex and development related issues
  - libocca.slack.com
- ALCF specific support
  - kris.rowe@anl.gov

Questions?