POSTER P5129 | CONTACT NAME Jungwon Kim: kimj@ornl.gov

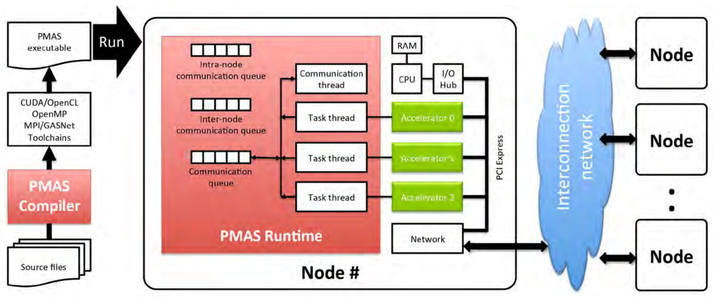
PMAS: A Unified Programming Model for Multi-Accelerator Systems

Jungwon Kim, Seyong Lee, and Jeffrey S. Vetter
Oak Ridge National Laboratory

OAK RIDGE National Laboratory | U.S. DEPARTMENT OF ENERGY | Office of Science

What is PMAS?

PMAS is a unified ***P***rogramming model for ***M***ulti-***A***ccelerator ***S***ystems.



The goal of PMAS is to provide an easy, efficient, and uniform way for programmers to program heterogeneous systems with multiple accelerators, in both single and multi-node configurations.

PMAS is a SPMD model of OpenACC. When a user launches a PMAS program, a number of program instances, called tasks, execute simultaneously in parallel.

The major contributions of PMAS:

- ***Portability.*** An accelerator program written in PMAS is portable across a single accelerator, multiple accelerators in a single node, and a distributed system.
- ***Programmability.*** PMAS is a directive-based, high-level programming model. It hides the underlying hardware architecture complexity and heterogeneity from programmers.
- ***Performance.*** Task implementation using lightweight user-level threads and the tight integration of low-level accelerator APIs and inter-node communication APIs in PMAS allows various architecture-specific optimizations.
- ***Scalability.*** Due to the autonomous task execution, good locality, and low runtime overhead, PMAS achieves good scalability on large-scale systems.

A simple example

The following code snippet illustrates a Jacobi code using PMAS.

```
#include <pmas.h>

double (*xlocal)[MAXN], (*xnew)[MAXN];
double diffnorm, gdiffnorm;
double temp[MAXN];
#pragma acc data localshared(temp)
...
void jocobi() {
  int myid = acc_get_task_num();
  int ntasks = acc_get_num_tasks();
  size_t size = ((MAXN / ntasks) + 2) * MAXN * sizeof(double);
  xlocal = (double(*)[MAXN]) malloc(size);
  xnew = (double(*)[MAXN]) malloc(size);
...
#pragma acc data create(xnew[0:(MAXN/ntasks)+2][0:MAXN]) \
                 copyin(xlocal[0:(MAXN/ntasks)+2][0:MAXN])
  do {
    if (myid < ntasks - 1)
      acc_mem_send(myid + 1, acc_deviceptr(xlocal[MAXN / ntasks]),
                   MAXN * sizeof(double), 0);
    if (myid > 0)
      acc_mem_recv(myid - 1, acc_deviceptr(xlocal[0]),
                   MAXN * sizeof(double), 0);
    ...
    #pragma acc kernels loop independent reduction(+:diffnorm)
    for (...) { ... }
    acc_mem_allreduce(&diffnorm, &gdiffnorm, 1, acc_double, acc_sum);
  } while (gdiffnorm > 1.0e-2)
}
```

Global variables annotated with *#pragma acc data localshared* directive are shared between tasks in the same node. All other data are private to each task.

The ID for each task and the total number of tasks in the whole system can be retrieved by *acc_get_task_num* and *acc_get_num_tasks* respectively.

#pragma acc data create copyin directive defines a data region, and it allocates device memory and copies data from host to device memory.

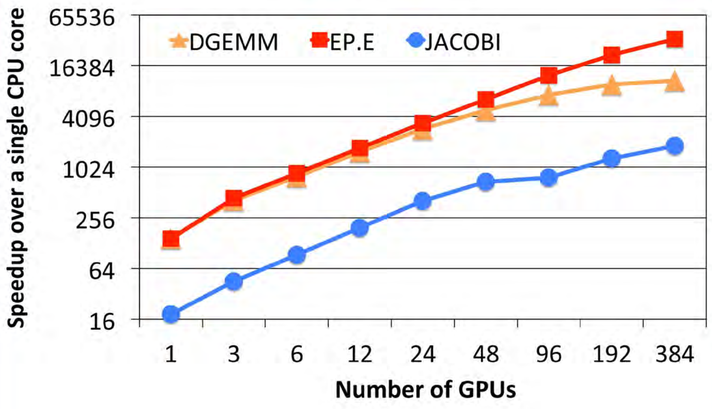
acc_mem_send and *acc_mem_recv* transfer data between two tasks. The data can reside in either host or device memory. *acc_deviceptr* returns the device pointer associated with a specific host address.

#pragma acc kernels identifies a kernel region to be offloaded on the accelerator.

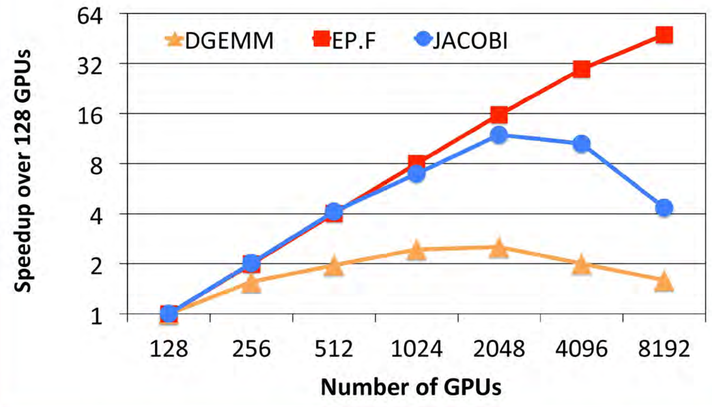
acc_mem_allreduce combines values from all tasks and distributes the results back to all tasks.

Preliminary evaluation

Keeneland
Ranked #162 - TOP500 (Nov 2014)
Georgia Institute of Technology
2x Intel Xeon E5 CPUs, 32GB RAM
3x NVIDIA Tesla M2090 GPUs
Mellanox FDR InfiniBand
264x Compute nodes



Titan
Ranked #2 - TOP500 (Nov 2014)
Oak Ridge National Laboratory
AMD Opteron CPU, 32GB RAM
NVIDIA Tesla K20x GPU
Cray Gemini interconnect
18,688x Compute nodes



This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research.
This manuscript has been authored by UT-Battelle, LLC, under Contract No. DE-AC0500OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for the United States Government purposes.
This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.