Abstract. With high performance systems exploiting multicore and accelerator-based architectures on a distributed shared memory system, heterogenous hybrid programming models are the natural choice to exploit all the hardware made available on these systems. Previous efforts looking into hybrid models have primarily focused on using OpenMP directives (for shared memory programming) with MPI (for inter-node programming on a cluster), using OpenMP to spawn threads on a node and communication libraries like MPI to communicate across nodes. As accelerators get added into the mix, and there is better hardware support for PGAS languages/APIs, this means that new and unexplored heterogenous hybrid models will be needed to effectively leverage the new hardware. In this paper we explore the use of OpenACC directives to program GPUs and the use of OpenSHMEM, a PGAS library for one-sided communication between nodes. We use the NAS-BT Multi-zone benchmark that was converted to use the OpenSHMEM library API for network communication between nodes and OpenACC to exploit accelerators that are present within a node. We evaluate the performance of the benchmark and discuss our experiences during the development of the OpenSHMEM+OpenACC hybrid program.

1 Introduction

New HPC systems are increasingly turning to accelerators to increase compute power while mitigating the rising cost of power [1]. For example, four of the top ten super computers use GPUs as their main devices to perform the majority of the computations. Oak Ridge National Laboratories Titan [2], a DOE leadership class machine, makes extensive use of GPUs, using one Nvidia Kepler GPU per node. Without a major breakthrough in technology, the future of the fastest super computers will consist of clusters with multiple cores and attached to specialized devices for accelerators, interconnects and I/O. Modern cluster nodes
have many different types of hardware that need to be exploited efficiently to make the maximum use of the provided resources. Current nodes have multiple sockets with attached memory, each socket has a CPU with multiple cores. On top of this the node usually has an attached accelerator, currently the most common one is the GPU. In addition the nodes are all connected with network hardware that provides better support for PGAS languages/libraries to allow them to communicate. This means that each node has three different major components that need to be programmed for. Each of these components has its own programming model with its own challenges. When these models are used together for hybrid programming models, new challenges arise, specially when one of the models deals with heterogeneous programming. In this paper we explore a new hybrid programming model OpenSHMEM+OpenACC.

OpenSHMEM is the result of standardizing several shmem libraries [3]. It is a one-sided communication library where individual processes do one-sided puts and gets, as compared to MPI that does synchronized send/receive between pairs of processes. This allows for data to be sent without having to wait on remote nodes to do communication. OpenACC is the result of standardizing compiler directives for accelerator programming sanctioned by the OpenACC organization. It allows for an OpenMP-like programming with support for incremental parallelism. This includes the ability to incrementally add directives to a code to program for GPUs, rather than having to do a considerable amount of code restructure to just start using an accelerator (like with the OpenCL standard). We use the NASA Advanced Supercomputing (NAS) Block Tri-diagonal (BT) Multizone benchmark [4] to evaluate our results. This benchmark is structured so that there are multiple zones that can be solved independently with the boundary values of each zone exchanged on each iteration, making it well suited to experimentation with heterogeneous hybrid programming model.

This paper is organized into 5 sections. In Section 2 we discuss the other research done on hybrid programming models and provide background information on the BT-MZ benchmark used, the OpenACC directives, and the OpenSHMEM library in Section 3. In Section 4 we discuss the implementation details of how OpenSHMEM and OpenACC are used together. The results are presented in Section 5. We discuss the platforms used and the timings from running BT-MZ on those platforms. In Section 6 we interpret the results we obtain and discuss the future paths for exploration in this hybrid programming.

2 Related Work

Hybrid models that explore shared-memory and distributed-memory programming have been researched over the last few decades. The idea is to exploit the strengths of the different models, including the in-node efficiency, memory savings, accelerator programming, and the scalability characteristics within a distributed memory system. The shared and distributed programming models models have been evolving separately and an attempt to unify them resulted in the creation of new languages and models such as the HPCS and PGAS languages (X10, Chapel, Fortress, UPC, etc). In terms of heterogenous programming, there