# Persistent Memory Exercises

# Practical 3

Adrian Jackson

## 1 Introduction

In this exercise we are going to investigate using the PMDK approach to programming B-APM. Firstly, we will just try using the memkind approach to use the memory as a large capacity (but slower than DRAM) memory space. Then we will alter STREAMs so that only the persistent memory is used and look at different performance impacts of when the persistent operations are undertaken.

Before starting this practical you should ensure you have already completed Practical 1 (https://github.com/NGIOproject/PMTutorial/blob/master/Exercises/practical1.pdf), which also has detailed instructions on accessing and using the system.

This practical is designed to give you hands on experience using PMDK to program B-APM including defining when and where persistence happens. It does involve programming in C, so if you are not confident in C programming, or programming in general, get in touch with the tutorial organisers for more guidance, or the solutions to the exercises so you can examine the approach taken.

## 2 Volatile Non-Volatile Streams

For this practical the goal is to change the STREAMs source code to utilise libmemkind to create the memory allocations that streams uses for its calculations. To exploit libmemkind you will need to use the following functions and header files:

```
#include <memkind.h>

memkind_create_pmem

memkind_malloc
```

You will also need to load the memkind module (`module load memkind`) and add this to your batch script to compile and run your modified application. Record the performance you get when using B-APM for STREAMs using memkind.

## 3 Persistent Streams

For this practical the goal is to change the streams source code to utilise PMDK to create the memory allocations that streams uses for its calculations. To exploit PMDK you will need to use the following functions and header files:

```
#include <libpmem.h>

pmem_map_file

pmem_persist
```

```
pmem_unmap
```

You will also need to load the PMDK module (`module load pmdk`) and to add this to your batch script to compile and run your modified application.

There are different choices that can be made on when and where to persist data within the STREAMs kernels, and which parts of the streams data to place in volatile and non-volatile memory. Try different persistent combinations and memory allocation locations to investigate impact on performance. You could consider the following approaches:

1. Persist after each write operation
2. Persist after each benchmark
3. Persist after all four benchmarks

Investigate each approach and evaluate the impact on performance. You can also investigate the performance when different arrays are in DRAM vs B-APM, i.e. only place one or two of the arrays in B-APM. What performance variation do you see?

# 4 Correct Persisting

To investigate correct persistence in applications create two application, one that writes data to persistent memory using PMDK and the other that reads the data. Ensure you persist memory at the end of the writing application and run your two applications so one consumes the data from the other and checks it is correct. Once you have done that correctness check, remove the persistence in the write application, compile, and re-run your workflow (the two applications together). Can the reader now correctly obtain the data is it expecting?