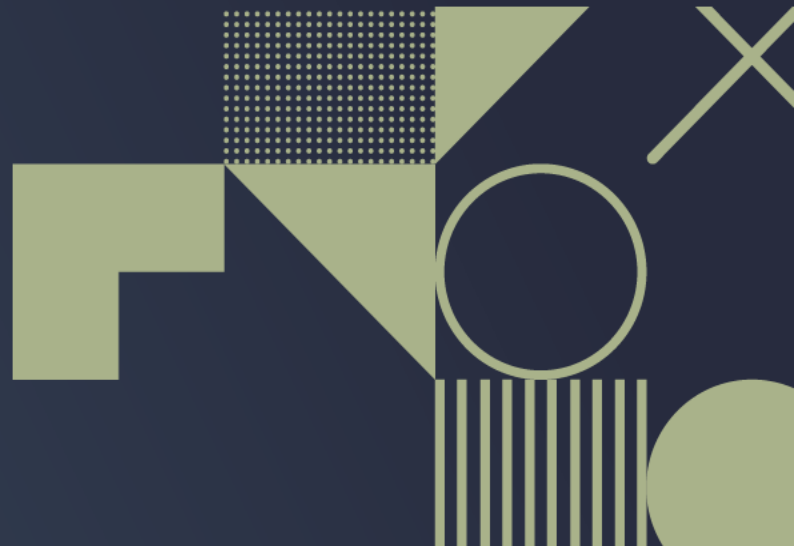
PRACTICAL PERSISTENT MEMORY PROGRAMMING: PMDK AND DAOS

Adrian Jackson

Johann Lombardi

Mohamad Chaarawi



TUTORIAL

Practical Persistent Memory Programming: PMDK and DAOS

Johann Lombardi
Mohamad Chaarawi

Adrian Jackson
(a.jackson@epcc.ed.ac.uk)

Aims

- Understand persistent memory hardware and software
- Understand the different aspects that can impact performance, and the shared/private nature of the resources
- Learn how to program persistent memory
- Learn about DAOS
- Get hands on with persistent memory hardware

intel

epcc

Aims cont.

- Understand data movement and think about application data requirements
- Understanding I/O and data movement, particularly persistence requirements, is hard
- Thinking about different ways you undertake I/O or storing data
- Move beyond bulk, block-based, I/O paradigms

Format

- Lectures and practicals
- Slides and exercise material available online:
  - https://github.com/NGIOproject/PMTutorial
  - Exercises will be done on remote machine (NEXTGenIO prototype)
  - You should have accounts for these, or contact us if you don't

Timetable

- 13.00 Introduction
- 13.10 Hardware, I/O and storage
- 13.30 Low-level persistent memory programming
- 14.00 Practical: Persistent memory programming
- 14.20 Persistent Thinking
- 14.30 DAOS and Persistent Memory Usage
- 15.00 Break
- 15.30 DAOS API
- 16.20 Practical: Hands on Session with DAOS
- 17.00 Summary and finish

Programming persistent memory

intel

epcc

Programming persistent memory

```
double *a, *b, *c;
pmemaddr = pmem_map_file(path, array_length,
                         PMEM_FILE_CREATE|PMEM_FILE_EXCL,
                         0666, &mapped_len, &is_pmem)
a = pmemaddr;
b = pmemaddr + (*array_size+OFFSET)*BytesPerWord;
c = pmemaddr + (*array_size+OFFSET)*BytesPerWord*2;

#pragma omp parallel for
for (j=0; j<*array_size; j++){
   a[j] = b[j]+scalar*c[j];
}
pmem_persist(a, *array_size*BytesPerWord);
```

intel

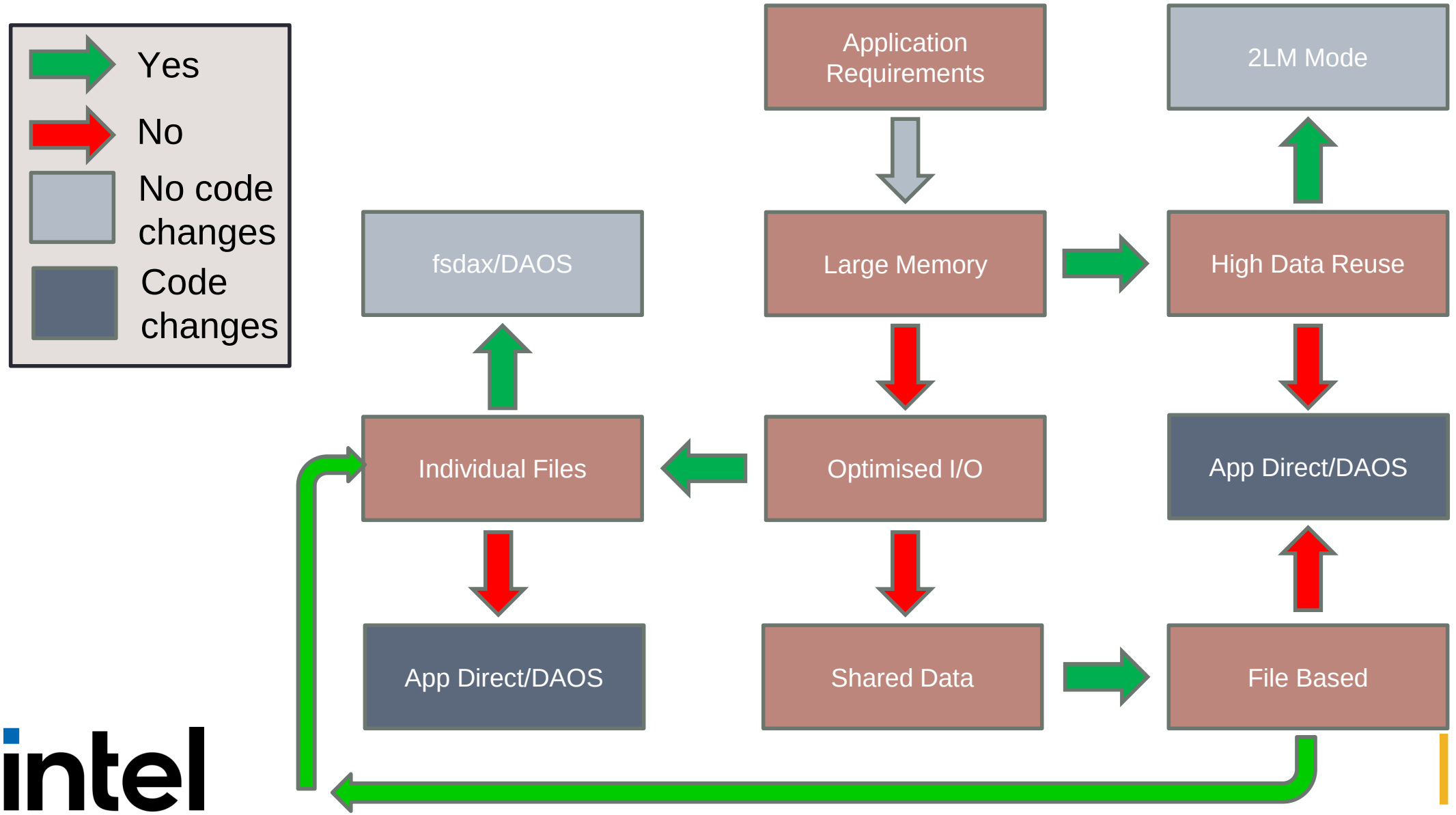
epcc

Programming persistent memory

intel

epcc

Programming Persistent Memory



intel

epcc

Programming persistent memory

- Design and performance considerations are the challenge
  - Programming the memory is easy
- Design for functionality
  - What is persistent, when is it persistent, what failures can you tolerate, etc..
- Design for performance
  - Memory size, I/O, data access costs, etc...
- Design for hardware configurations
  - NUMA, filesystems, storage, etc...

Summary

- Please don’t hesitate to ask questions!
- We are doing practicals
  - But if you’re not confident in programming we have other options