



THE UNIVERSITY *of* EDINBURGH

Edinburgh Research Explorer

## **xDSL: A common compiler ecosystem for domain specific languages**

### **Citation for published version:**

Brown, N, Grosser, T, Fehr, M, Steuwer, M & Kelly, P 2022, 'xDSL: A common compiler ecosystem for domain specific languages', Supercomputing 2023, Denver, United States, 12/11/23 - 17/11/23.

### **Link:**

[Link to publication record in Edinburgh Research Explorer](#)

### **Document Version:**

Peer reviewed version

### **General rights**

Copyright for the publications made accessible via the Edinburgh Research Explorer is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

### **Take down policy**

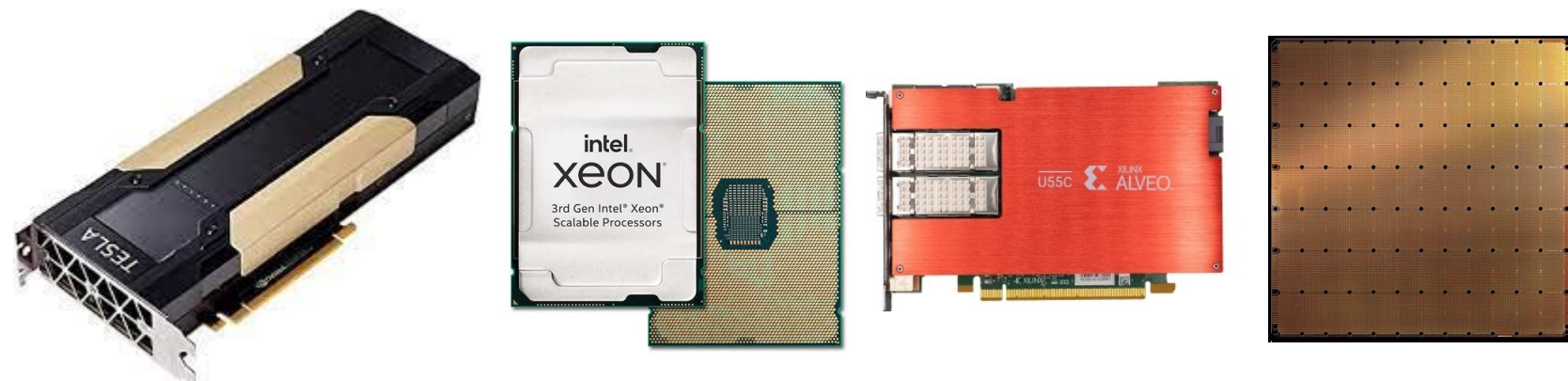
The University of Edinburgh has made every reasonable effort to ensure that Edinburgh Research Explorer content complies with UK legislation. If you believe that the public display of this file breaches copyright please contact [openaccess@ed.ac.uk](mailto:openaccess@ed.ac.uk) providing details, and we will remove access to the work immediately and investigate your claim.





## What are Domain Specific Languages (DSLs)?

There is an explosion of hardware architectures for HPC, and as we move into the exascale era a key challenge is how to fully exploit such complex, highly parallel supercomputers.



DSLs provide a separation of concerns, where domain specific abstractions enable programmers to concentrate on their application logic whilst providing a rich amount of information upon which the toolchain can determine tricky low-level decisions around parallelism.

The term *language* is a misnomer, tools and frameworks that provide domain specific **abstractions** would be better

**Arguably, DSLs are the only way in which we can effectively program and exploit exascale supercomputers**

## Why are DSLs not ubiquitous in HPC?

A major disadvantage is in the siloing of compiler infrastructure, where DSLs tend to share very little or no infrastructure between them at the toolchain level.

**This siloing of toolchains results in:**

- ✗ Uncertainty around long term maintenance
- ✗ Reinventing the wheel
- ✗ Considerable effort needed to develop a DSL
- ✗ Limited opportunities for DSLs to target multiple domains
- ✗ Limited third-party tools
- ✗ Considerable effort for new architectures

**We must solve the challenges around siloing in order for DSLs to become more widespread**

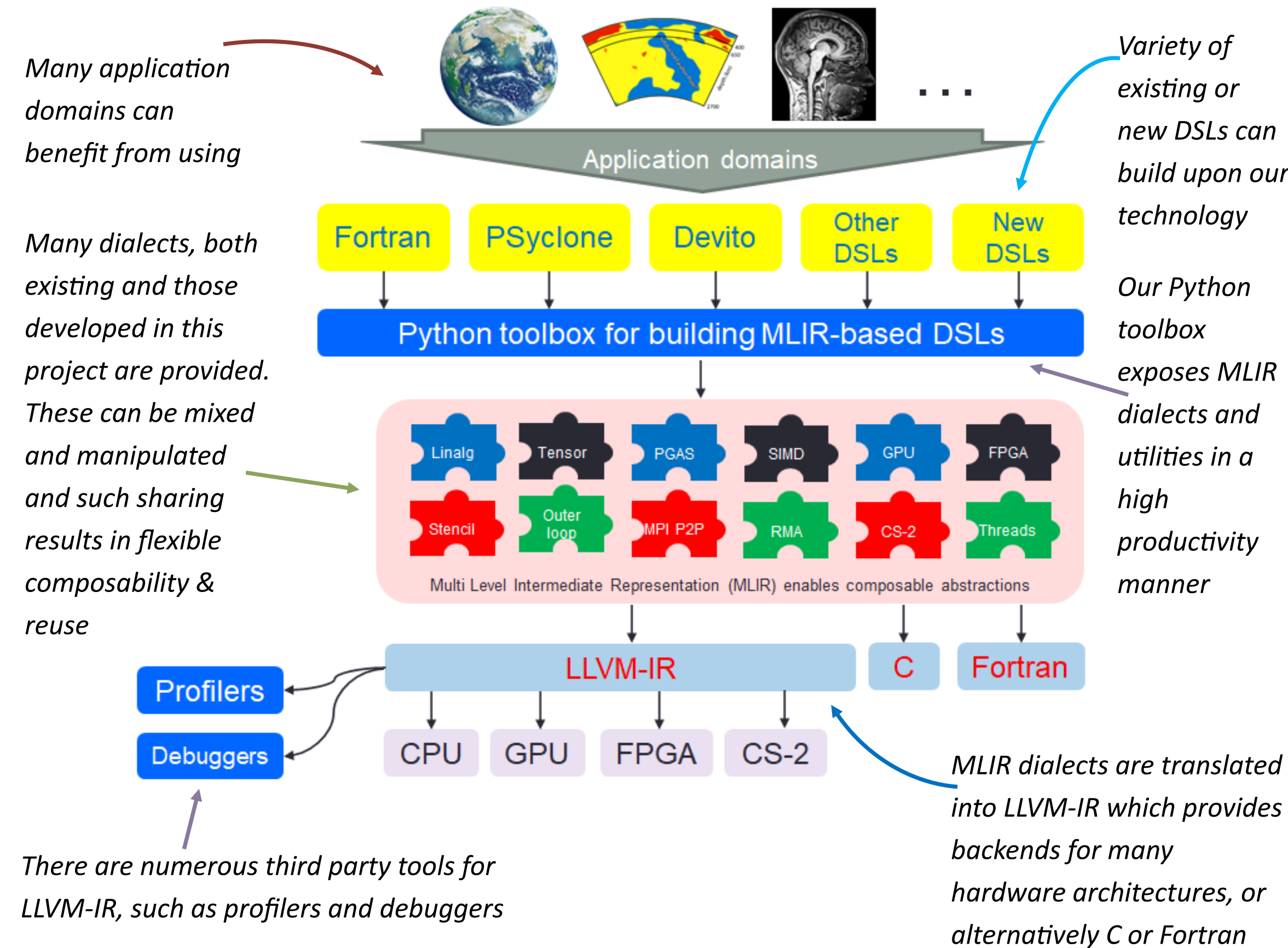
**Hence our vision: A DSL being a thin layer atop an existing, mature ecosystem with a wealth of third party tools**

## 30 second summary

- Domain Specific Languages (DSLs) have great potential.
- Underlying toolchains are often siloed and share very little or no infrastructure.
- We are developing a Python ecosystem so DSL developers can write a thin-layer on-top of existing, well supported, MLIR/LLVM compiler infrastructure.

## What is our xDSL ecosystem?

*Python toolbox based upon MLIR, integration with MLIR & series of HPC dialects*



There are numerous third party tools for LLVM-IR, such as profilers and debuggers

### Our ecosystem provides:

- **Composability**, where DSL owners choose what parts of our ecosystem to leverage
- **Performance** because dialects and backends are developed by experts
- **Interoperability** between DSLs
- **Productivity** for both the application and DSL developers due to code reuse
- **Code reuse** of toolchain infrastructure
- **Portability** for application codes and DSLs across architectures.
- **Longevity** of DSL compiler technology as we build upon LLVM and MLIR

## Building on LLVM and MLIR

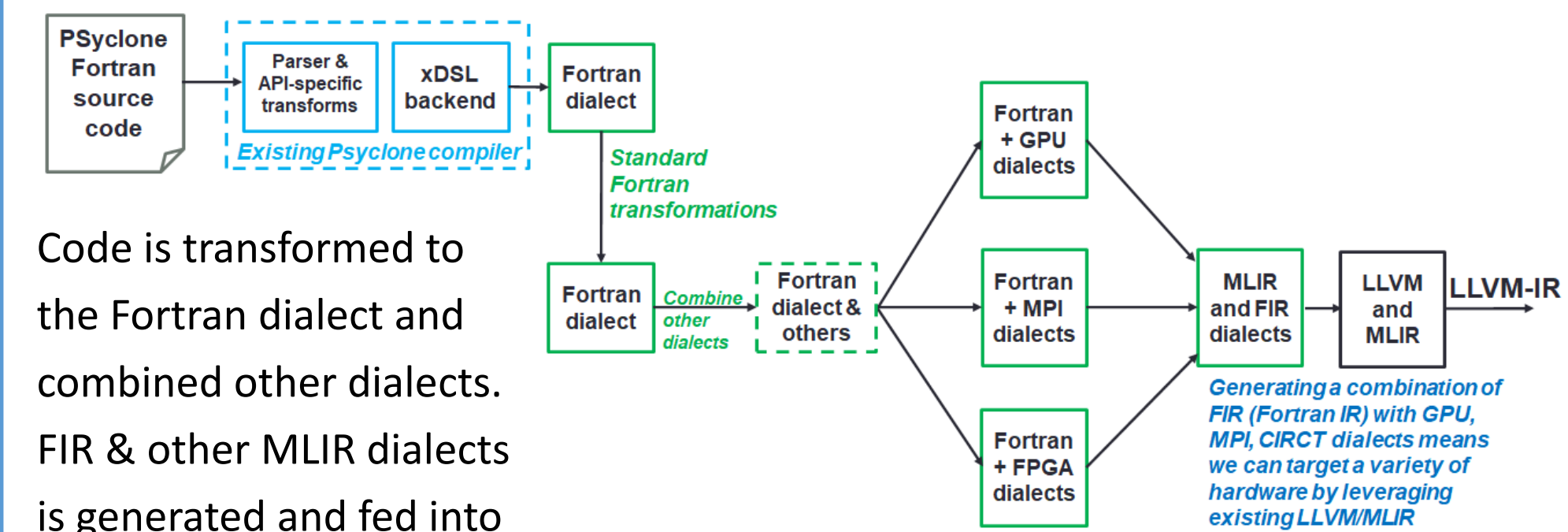
LLVM is a collection of common compilation tools and infrastructure. There are numerous LLVM backends available for different hardware, targeted via LLVM-IR.



MLIR enables representing and mixing dialects of intermediate representations and abstractions, thus providing easier integration, reuse, and optimisation. However, currently written in C++ there is a fairly steep learning curve that our Python toolbox looks to address.

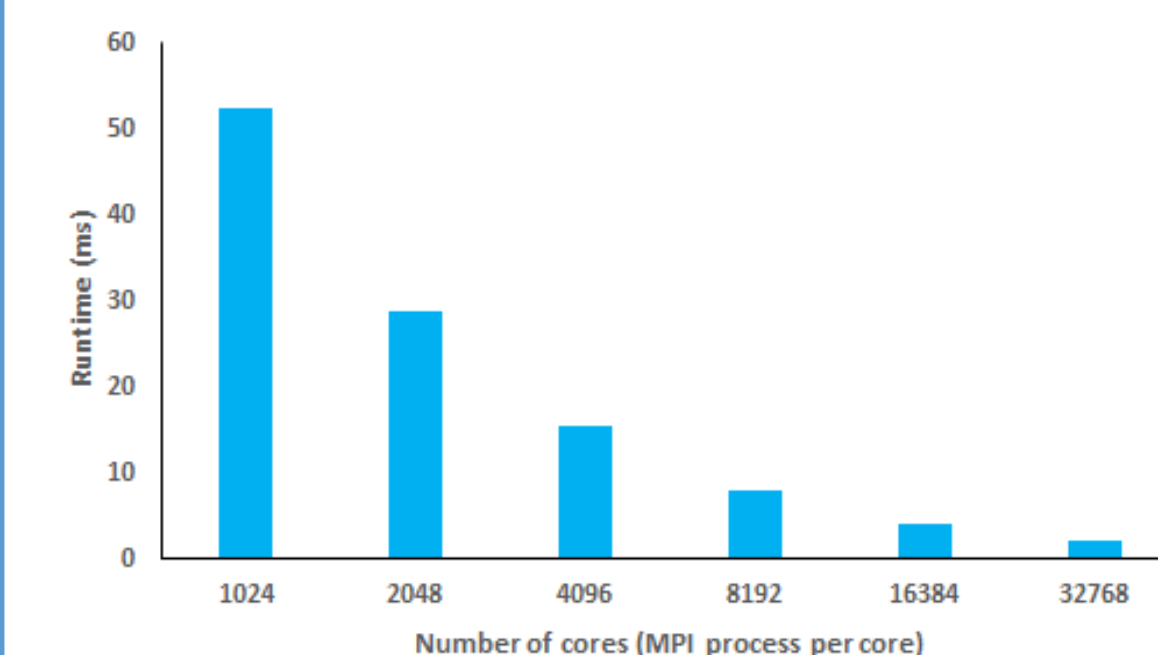
## Case study: Integrating xDSL with a Fortran DSL

PSyclone is Fortran-based DSL developed by STFC & the Met Office and used for weather and climate codes including LFRic & NEMO



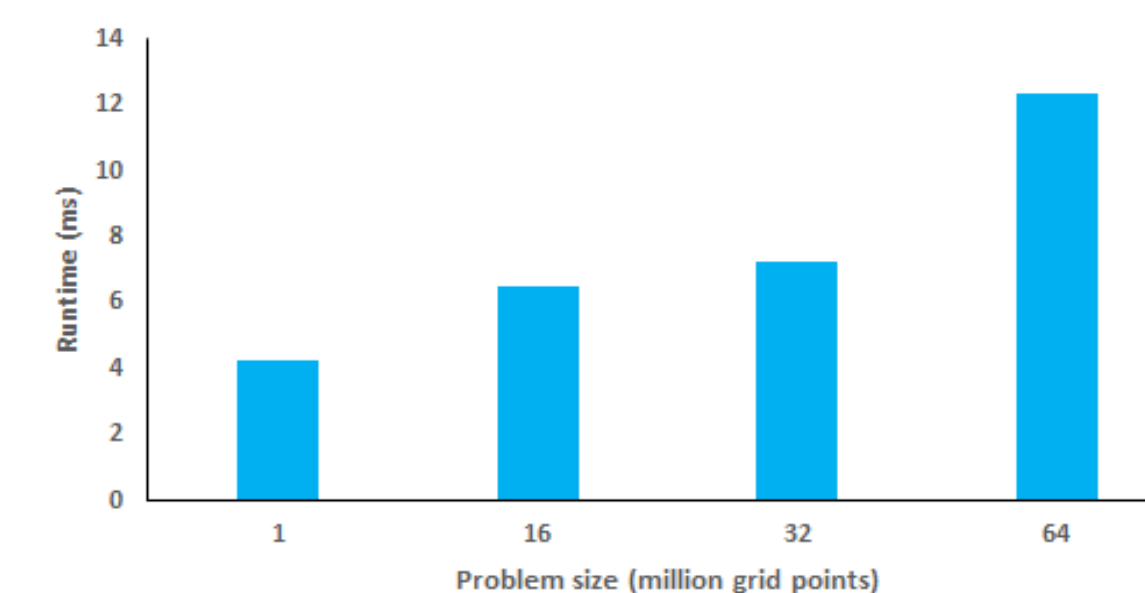
Code is transformed to the Fortran dialect and combined other dialects. FIR & other MLIR dialects is generated and fed into LLVM/MLIR tooling

Strong scaling performance on ARCHER2



Fortran is an example, xDSL can be used with many different languages. For instance the Devito DSL which is Python-based is also using xDSL, sharing many of the underlying dialects and transforms

Problem size scaling performance on single V100 GPU



Using PW advection scheme with 137 billion global grid points on ARCHER2 with 128-core AMD Rome CPU nodes. Running on Cirrus V100s for GPUs.



## Funded by ExCALIBUR

The UK ExCALIBUR program address the challenges and opportunities offered by computing at the exascale and aims to deliver the next generation of HPC simulation software and tooling

<https://excalibur.ac.uk/>