Science and Technology Facilities Council

Hartree Centre

PSyclone, a source-to-source transformation and optimisation tool for Fortran

Sergi Siso[1], Rupert Ford[1], **Andrew Porter**[1], Chris Dearden[1], Iva Kavcic[2], Chris Maynard[2], Joerg Henrichs[3], Aidan Chalk[1], Nuno Nobre[1], Mike Bell[2] …

[1]STFC Hartree Centre
[2]UK Met Office
[3]Australian Bureau of Meteorology

ExCALIBUR Workshop, 17th-18th October, 2024

Overview

1. Motivation
2. PSyclone in ExCALIBUR
    1. Marine-systems models
    2. OpenMP Tasking
    3. [xDSL]
3. Conclusions

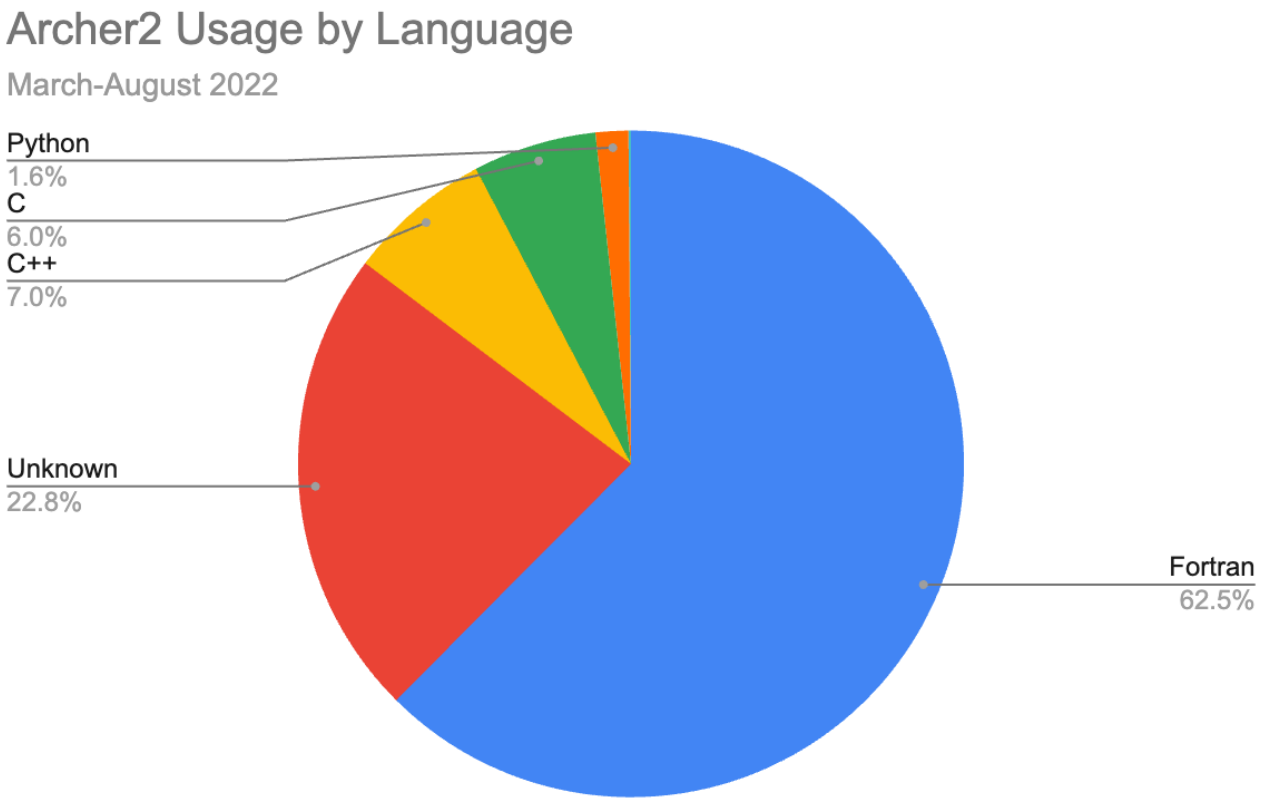
1. Motivation

HPC is heterogeneous (and diverse)

Top 500 June ‘24 list

#	Name	Processor	Linpack PFlop/s
1	Frontier	AMD EPYC Milan 64c AMD Instinct MI250X	1,207
2	Aurora	Intel Xeon Max 9470 52c Intel Data Center GPU Max	1,012
3	Eagle	Intel Xeon Platinum 8480C 48c Nvidia H100	846
4	Fugaku	Fujitsu A64FX 48c	442
5	Lumi	AMD EPYC Milan 64c AMD Instinct MI250X	379
6	Alps	Nvidia Grace 72c Nvidia GH200	270
7	Leonardo	Intel Xeon Platinum 8358 32c Nvidia A100 SMX4	241

Many HPC applications still use Fortran



Code	Language	Percentage use
VASP	Fortran	27.29%
Unidentified	Unknown	22.35%
CP2K	Fortran	6.28%
GROMACS	C	4.53%
CASTEP	Fortran	4.03%
Met Office UM	Fortran	3.10%
SENGA	Fortran	2.92%
Nektar++	C++	2.81%
NEMO	Fortran	2.46%
LAMMPS	C++	2.40%
PDNS3D	Fortran	1.89%
iIMB	Fortran	1.71%

Source: https://cpufun.substack.com/p/is-fortran-a-dead-language - Jim Cownie
https://www.archer2.ac.uk/support-access/status.html#:~:text=0.0-,Historical%20usage%20data,-Period

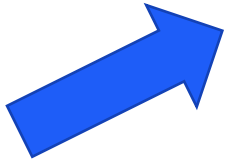
Collaborative development

- Some applications have many contributors from multiple institutions:
  - Run on different systems in each institution. Hard to maintain multiple implementations. It must be portable.
  - Contributors from different areas of expertise. Productivity, readability, maintainability are essential for the sustainability of the project.
  - Millions LOCs of FORTRAN (validated long-standing code).
  - Rewriting existing applications in a GPU-centric programming model is a challenging proposition.

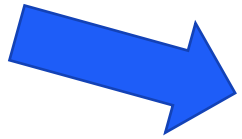
Fortran

```
A = B + maxval(C(:,3:8))
```

- High-level array notation
- Array intrinsics
- Slices
- Pure, Elemental
- Where, Masks
- Non-aliasing semantics



Expressive syntax



Good for optimising compilers

Fortran vs heterogeneous Fortran

```
A = B + maxval(C(:,3:8))
```

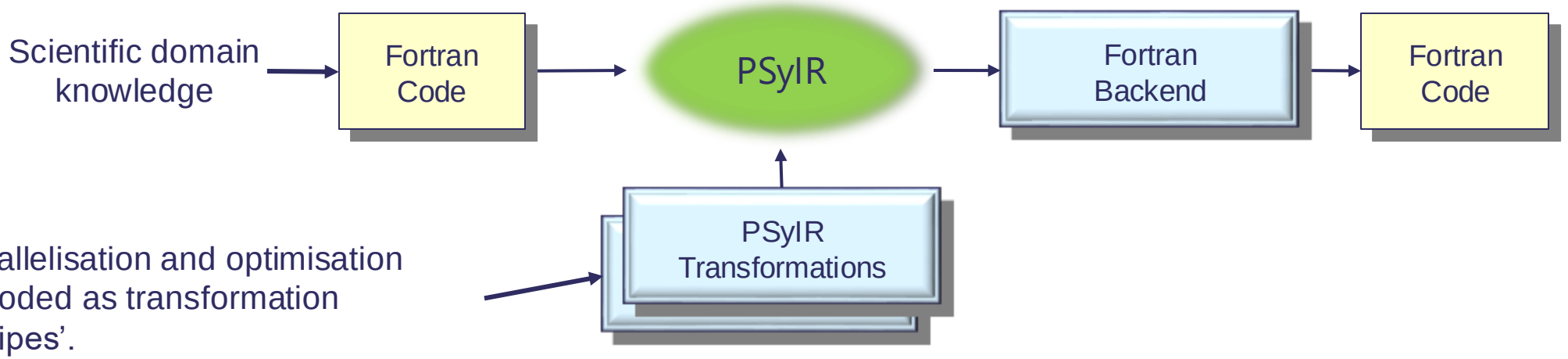
```
!$omp map data(from: B, C)
tmp = -huge(C)
!$omp target
!$omp loop collapse(2) reduction(tmp:max)
do i=1, N
    do j=3, 8
        tmp = max(tmp, C(i, j))
    end do
end do
!$omp end loop
!$omp loop collapse(2)
do i=1, N
    do j=1, M
        A(i,j) = B(i,j) + tmp
     end do
end do
!$omp end loop
!$omp end target
!$omp map data (to: A)
```

- Loses terse notation
- Harder to maintain, changing a line easily breaks directives

Is metaprogramming a solution?

- Start from a single-source science description in Fortran
- At build time generate the required specialised syntax to target a particular platform / programming model / set of parameters.

PSyclone (BSD 3-clause)
https://github.com/stfc/PSyclone



Separation of concerns

Perf portability: diff. recipes for each architectures

Visible "*readable*" output Standard debugging/profiling

Standard output composable w. other tools and vendor compilers

2. PSyclone for Marine Systems Models

Excalibur Marine Systems (‘nemo’) Design

Science and Technology Facilities Council
Hartree Centre
Met Office
National Centre for Atmospheric Science
NATURAL ENVIRONMENT RESEARCH COUNCIL
NVIDIA
National Oceanography Centre

NEMO Ocean Model

- **N**ucleus for **E**uropean **M**odelling of the **O**cean (https://www.nemo-ocean.eu/)
- Written in Fortran (120 kLOCs) with MPI
- NEMO is intended to be a flexible tool for studying the ocean and its interactions with the other components of the earth climate system (atmosphere, sea-ice, biogeochemical tracers) over a wide range of space and time scales.
- The NEMO Consortium comprises five European institutes:

Fortran with PSyclone: NEMO loop example

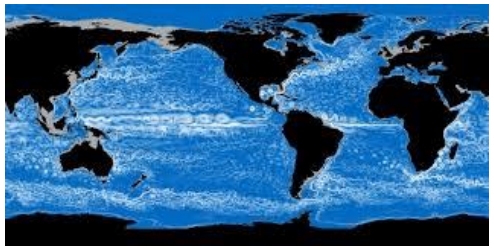
```
do jk = 1, jpkm1, 1
  zun(:,:,jk) = e2u(:,:) * e3u_n(:,:,jk) &
              & * (un(:,:,jk) + usd(:,:,jk))
enddo
```

```
for subroutine in psyir:

    for assign in subroutine.walk(Assignment):
        ArrayRange2LoopTrans.apply(assign)

    for loop in subroutine.walk(Loop):
        try:
            OMPLoopTrans().apply(loop)
            directive = loop.ancestor(Directive)
            OMPTargetTrans().apply(directive)
        except TransformationEror as err:
            print("Loop not accelerated:", err)
```

```
!$omp target
!$omp loop collapse(3)
do jk = 1, jpkm1, 1
  do idx_6 = LBOUND(zun, dim=2), UBOUND(zun, dim=2), 1
    do idx_7 = UBOUND(zun, dim=1),LBOUND(zun, dim=1), 1
      zun(idx_7,idx_6,jk) = e2u(idx_7,idx_6) * e3u_n(idx_7,idx_6,jk) &
                          & * (un(idx_7,idx_6,jk) + usd(idx_7,idx_6,jk))
    enddo
  enddo
enddo
!$omp end loop
!$omp end target
```

Can also contain domain specific logic and whole program optimisation!

NEMO 4 PSyclone-accelerated for GPUs

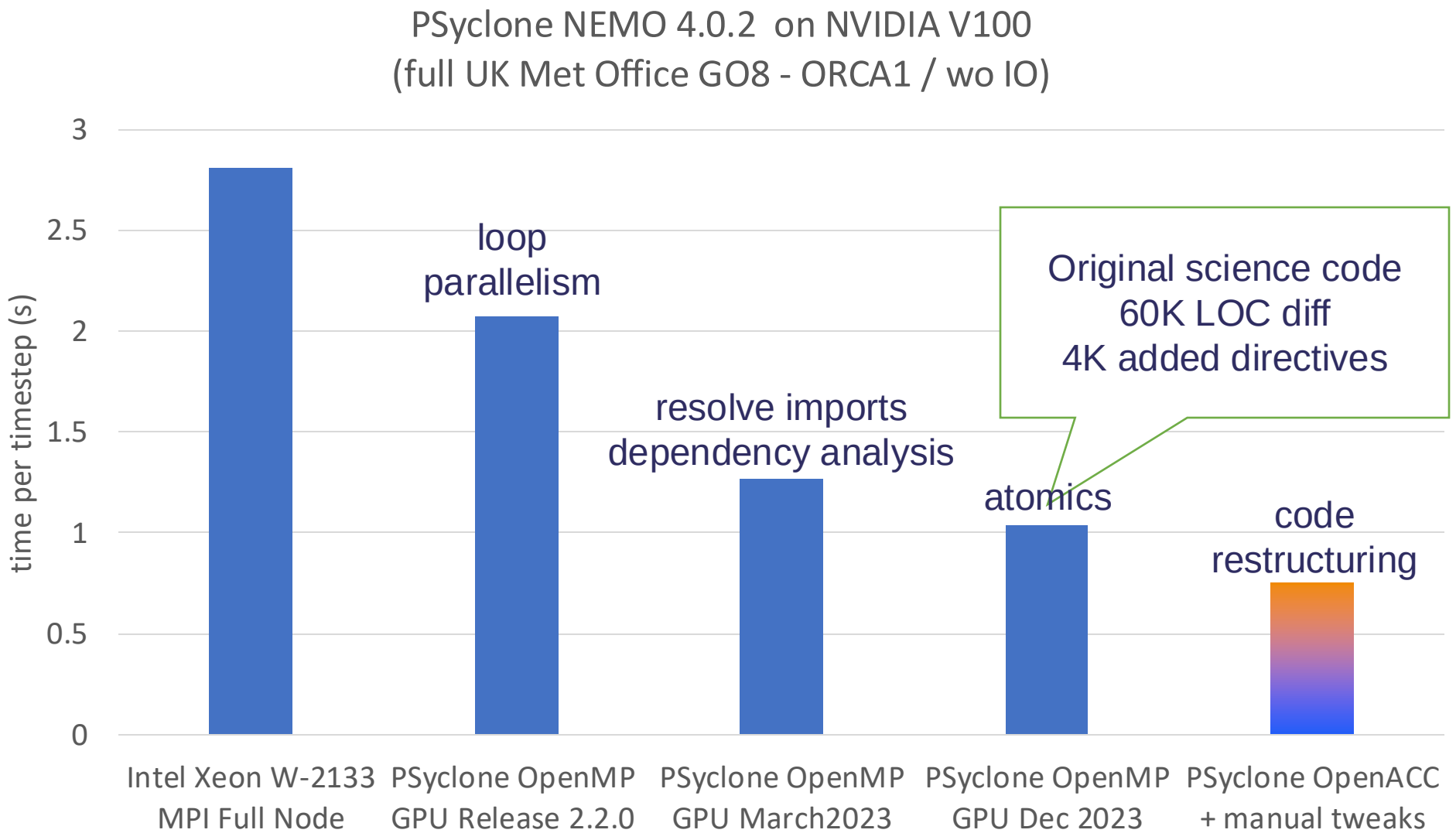


CPU:
Intel Xeon W-2133

GPU:
NVIDIA V100

Blue: original source
Orange: manual opt

Transformation:
~200 LOCs



Science and Technology Facilities Council
Hartree Centre

PSyclone for code transformation

- Provide performance portability to existing science code, incremental development.
- Successful at porting data-parallel, loop-centric code that already has MPI to GPUs.
- Performance portability can be fragile to code changes
  (DSL improves on this but not the subject of ExCALIBUR)

2. OpenMP Tasking in PSyclone

Excalibur Cross-Cutting Theme on Tasking

Science and Technology Facilities Council
Hartree Centre

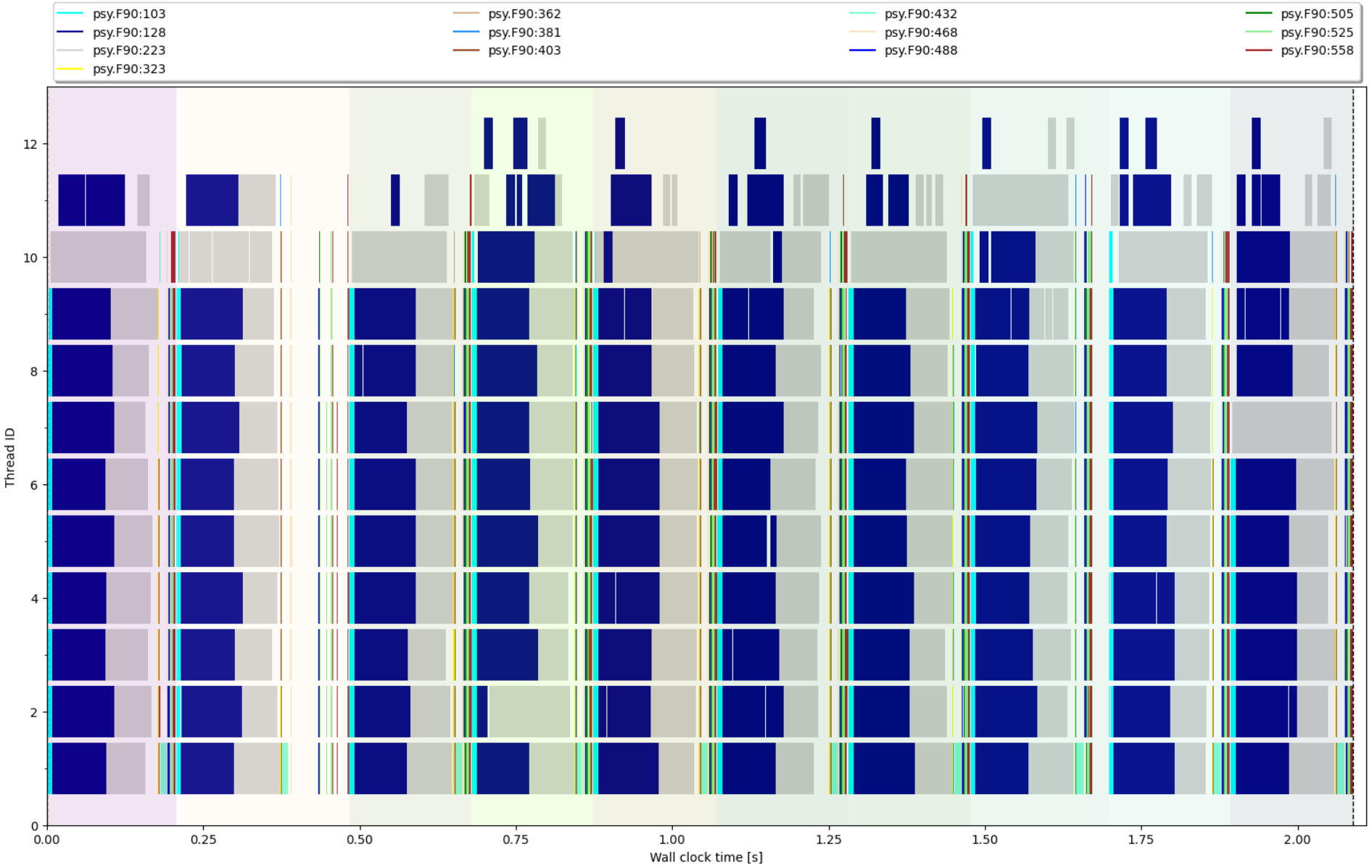
OpenMP tasking in PSyclone

- Goal: Can PSyclone add tasking into the generated code.
  - Primarily focused on the code transformation frontend in this project
- Strategy: Split loops into chunks, parallelise inner loop with tasks and dependencies.

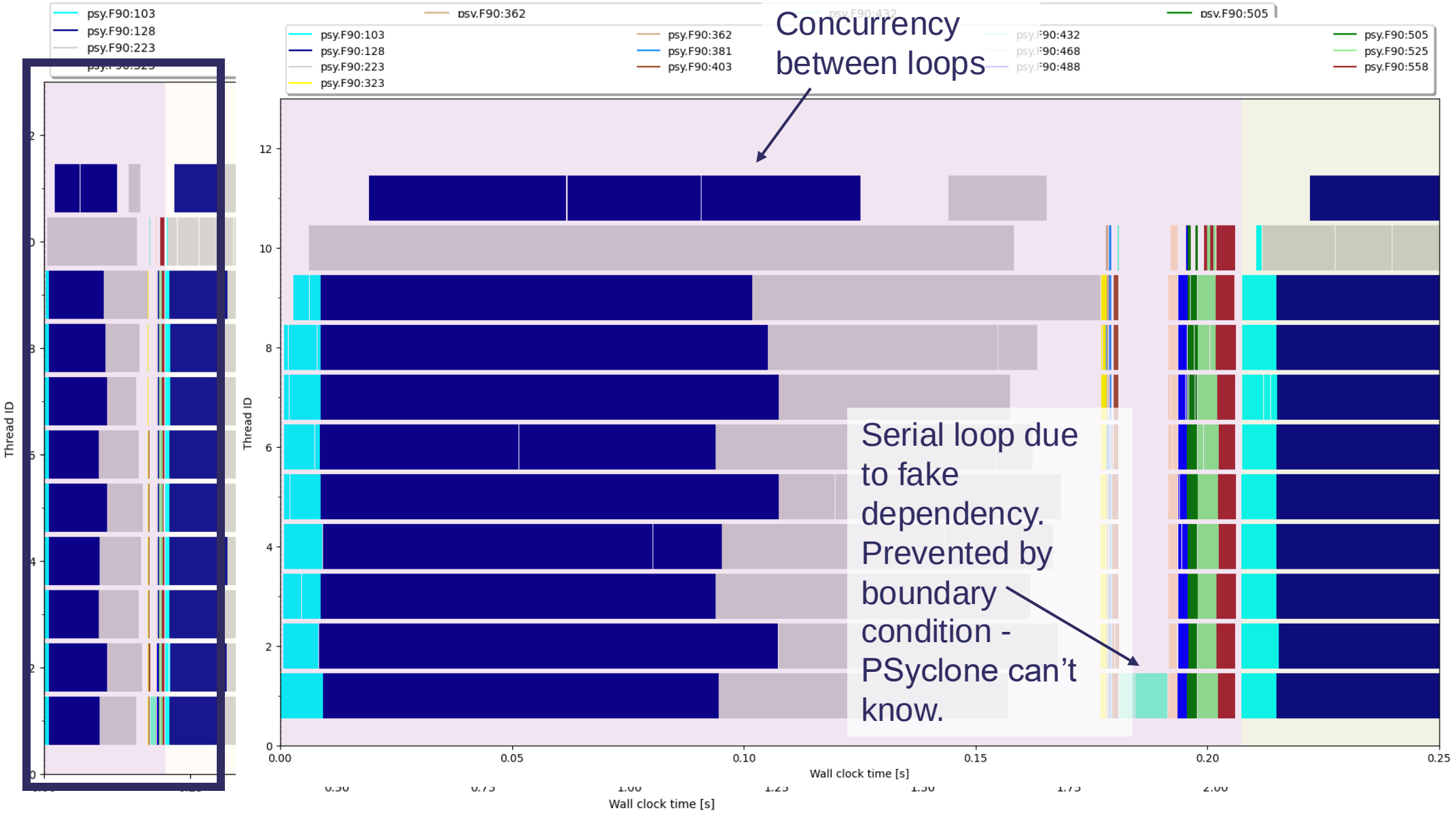
```
Do i = 1, N, 32
    !$omp task depend(.......)
      do j = i, i + 32, 1
```

- Challenges:
  - Dependency analysis to generate the depend clause(s).
  - Analysis of generated clauses to ensure correctness (`a(1:32)` and `a(2:31)` are *not* dependent in OpenMP 5.1 spec).

NemoLite2D Benchmark breakdown



NemoLite2D Benchmark Results - zoom



OpenMP Tasking Conclusions

- PSyclone is now capable of adding OpenMP tasking with dependencies into some code paths.
  - Full support for LFRic was too complex for this project's timeline.
- Performance can be competitive with traditional OpenMP looping code.
  - To be worthwhile, need a case with dependent loops with load imbalance, e.g. a lot of boundary condition sections.
- Some of the developments and improvements will benefit all future PSyclone development:
  - PSyclone may use a similar but simpler dependency analysis approach to launch `target` regions with `nowait` and add appropriate `taskwait` barriers into the code to increase GPU utilisation.

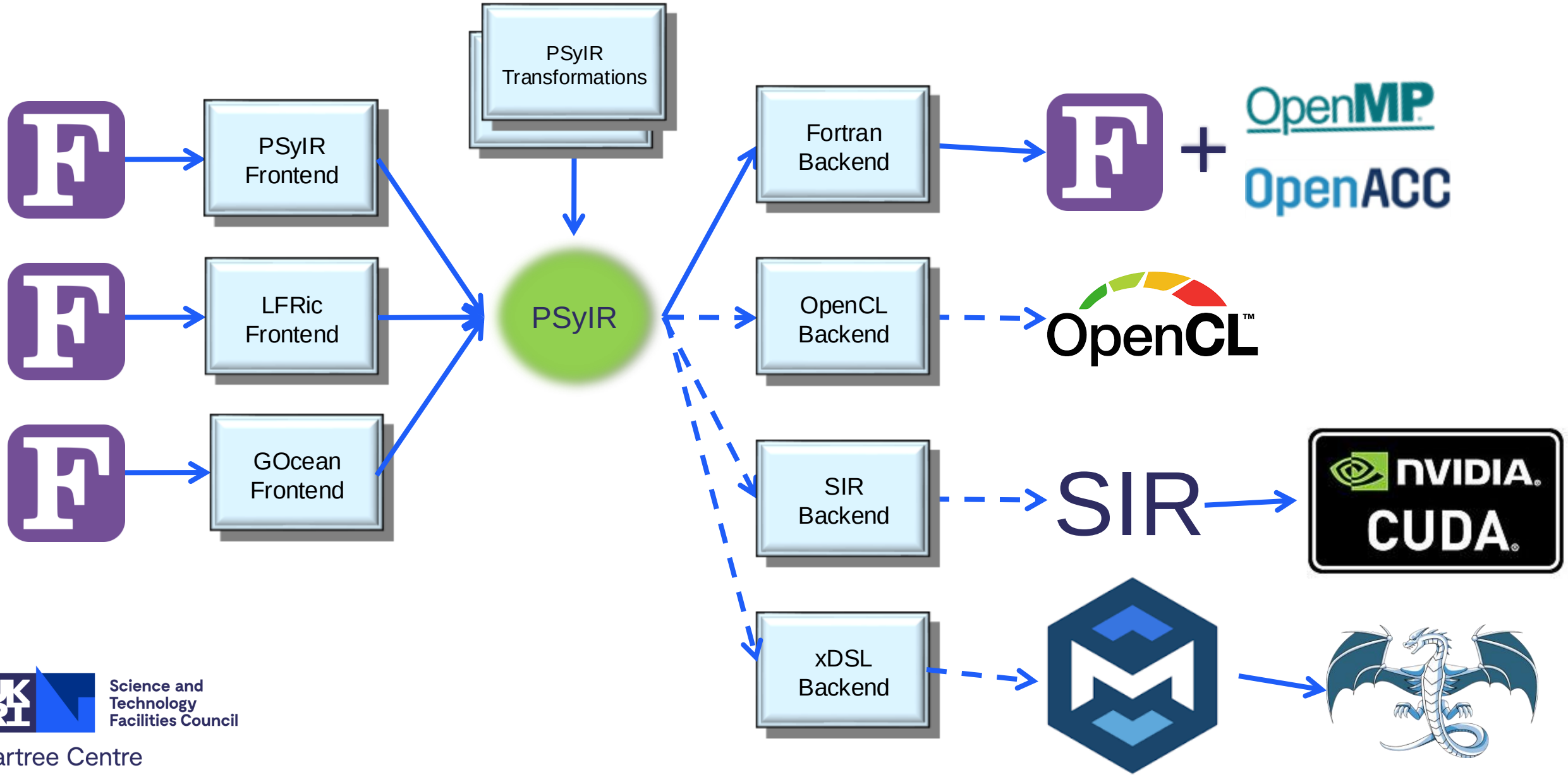
3. [PSyclone and xDSL]

Excalibur Cross-Cutting Theme on DSLs

Multi-Level Intermediate Representation

- Open-source compiler infrastructure, originally developed by Google
- “multi-level”: able to define multiple dialects and progressively convert towards machine code. This allows MLIR to retain information at a higher level of abstraction, which enables more accurate analyses and transformations.

PSyclone & xDSL



Science and Technology Facilities Council
Hartree Centre

Takeaways

- PSyclone is a Fortran source-to-source compiler that can be used with existing science code and to write DSLs.
- Provides separation of concerns and a tool for HPC experts.
- Support for
  - OpenMP threading, tasking and offload
  - OpenACC offload
  - Generic code transformations (inlining, hoisting, intrinsic replacement)
- Used with production/full configurations:
  - LFRic (multi-node parallelism for UK Met Office's atmospheric model)
  - NEMO (integrated in the build system and GPU demonstrator)
- Ongoing work on generalising the code-transformation approach and improving GPU offloading capabilities, especially within NG-Arch.

Science and Technology Facilities Council

Hartree Centre

Thank you

sergi.siso@stfc.ac.uk

andrew.porter@stfc.ac.uk

aidan.chalk@stfc.ac.uk