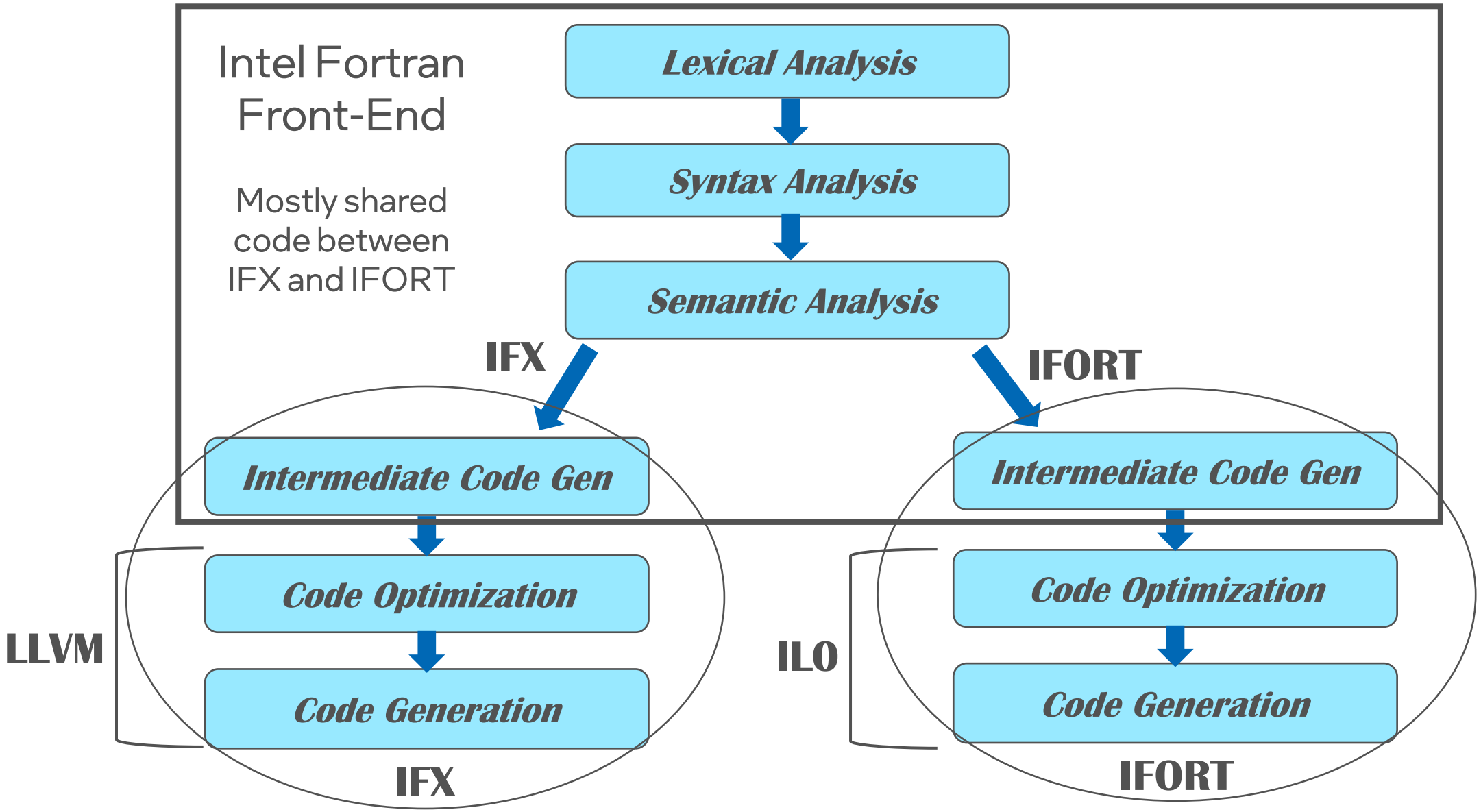
Creating
World
Changing
Technologies

Intel Fortran Compiler 2023

Getting Started & Porting from IFORT to IFX

intel®

How IFX Relates to IFORT



Important High-Level Understanding of IFX

Although both IFORT and IFX use the same language parser (Fortran Front End)
EVERYTHING after that is DIFFERENT

- Key takeaway
  - Optimization, vectorization, optimization reports, inlining, unrolling, interprocedural optimization, profile guided optimization, floating point control, code generation is different. SO …
- Examine the compiler options you are using
  - Remove most of the “exotic” performance options
  - Start with a simple subset like –`O2 –xhost`

Be aware that the default, out of box optimization used by ifx is aggressive, just like ifort

Specify the following options with ifx to turn off default optimizations:

- On Linux: **-O0 -fno-fast-math**
- On Windows: **/O0 /fp:precise**

Compiler Options for IFX First Use

Consider these safe options.
Use these to validate IFX correctness with your application before moving to optimized code tests.

	ifx Linux and ifx Windows
Disable optimization	-O0 /O0
Check for compile-time warnings	-warn all /warn:all
Runtime checks	-check all /check:all
Print stack traceback on crash	-g -traceback /debug:full /traceback
Create symbols for debugging	-g /debug:full
Turn off default OpenMP SIMD or !dir$ simd	-qno-openmp-simd -no-simd /Qopenmp-simd- /simd-
Obey Fortran semantics for expressions	-standard-semantics. /standard-semantics
Use precise floating-point settings	-fp-model=precise /fp:precise

Common Optimization Compiler Options

	Linux* ifx (ifort)
Disable optimization	-O0
Optimize for speed (no code size increase)	-O1
Optimize for speed (default)	-O2
High-level loop optimization	-O3
Create symbols for debugging	-g
Multi-file inter-procedural optimization	-ipo translates to –flto=full, use llvm-ar to create static libraries
Profile guided optimization (multi-step build)	-fprofile-generate (-prof-gen) -fprofile-use (-prof-use)
Optimize for speed across the entire program ("prototype switch")	-fast is same as "-ipo -O3 -static -fp-model fast" (-ipo -O3 -no-prec-div –static -fp-model fast=2 -xHost)
Recognize OpenMP directives	-qopenmp or -fiopenmp (-qopenmp)

IFX Performance Options

Option	Description/Use
-Ofast	Sets compiler options: -O3 -no-prec-div -fp-model fast=2
-flto[=[full | thin]]	Enables whole program link time optimization (LTO). -flto by itself sets flto=full. Thin see https://clang.llvm.org/docs/ThinLTO.html
-align arrayNbyte	N byte data alignment, use 32 on AVX2 or 64 on AVX-512
-ffast-math	Compatibility ICX macro option, same as IFX -fp-model=fast=2
-funroll-loops	Compatibility ICX option Unroll loops, same as IFX -unroll
-nostandard-realloc-lhs	Determines whether the compiler uses the current Fortran Standard rules or the old Fortran 2003 rules when interpreting assignment statements.

NOTE -nostandard-realloc-lhs should be used with caution. Assumes F2003 rules that require the LHS to be allocated AND with the correct shape to hold the RHS. If not, segfault or corrupted data. See the Fortran Developer Guide and Reference for more information

IFX Essentials: Options Support

- NEW: -qopenmp-simd is ON by default at –O1 and above
  - Turn off with `–qno-openmp-simd` or `/Qopenmp-simd-`
- IFORT options that are implemented are accepted quietly (no message)
- IFORT options that are not implemented generate a warning
  - `ifx: command line warning #10148: option '-simd' not supported`
- `ifx -qnextgen-diag` or `ifx /Qnextgen-diag`
  - Prints a long list of IFORT options TO BE supported
  - And prints a long list of IFORT options that are REMOVED

IFX -x and Intel Optimizations

- Classic compiler IFORT performs Intel-specific optimizations at -O2. Additional vectorization optimizations done if `-x` used.
- *LLVM compilers IFX will only do Intel specific optimizations with –x or –ax options*
  - Without `-x` or `-ax` you get default LLVM optimizations and vectorization
  - IMPLICATION: You ONLY get Intel optimizations and performance with IFX, if and only if, you use `-x` or `-ax` options.
    - `-O2` is NOT enough with IFX
    - `-xhost` tunes for the computer where the compile is done
- New! ifx -xsapphirerapids

Notes on Intel® AVX-512 Processor Targeting

- `-x` and `-ax` SUGGESTION or REQUEST to the compiler, not imperative
- IFX sometimes will use AVX2 instead of AVX-512
- Use –mprefer-vector-width=512

  ifx –mprefer-vector-width=512 …

  Same as IFORT compiler: –qopt-zmm-usage=high

Example:

ifx –xsapphirerapids –mprefer-vector-width=512

Optimization Report

- `-qopt-report[=n]` tells the compiler to generate an optimization report
  - **ifx has three levels of reports. n=3** is the max level
  - Includes Loop Optimizations, OpenMP parallelization and Register Allocation messages

- `-qopt-report-phase[=list]`
  specifies one or more optimizer phases for which optimization reports are generated.
  - loop: the phase for loop nest optimization
  - vec: the phase for vectorization
  - par: the phase for auto-parallelization
  - all: all optimizer phases

- `-qopt-report-filter=string`
  specifies the indicated parts of your application and generate optimization reports for those parts of your application.

Under development

More Optimization Report

- `-qopt-report[=n]` also creates a YAML file
  - Contains additional optimization information from LLVM
  - Use opt-viewer.py to create html files from the YAML file (available at llvm.org)
  - Information applies to host only

Interprocedural Optimization

ifort

-ip	Only between modules of one source file
-ipo	Modules of multiple files/whole application

ifx

-ipo (mapped to -flto)	Link Time Optimization in IFX

Without IPO

Compile & Optimize → file1.f90

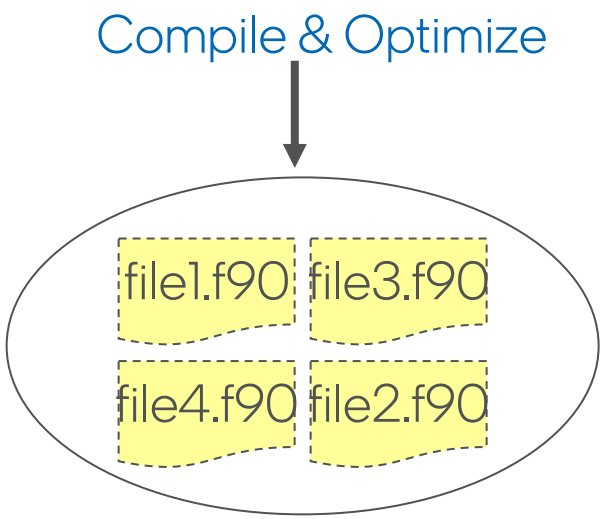
Compile & Optimize → file2.f90

Compile & Optimize → file3.f90

Compile & Optimize → file4.f90

Add -ipo to both compiling and linking steps

With IPO



Link Time Optimization (LTO): tinyurl.com/clang-lto

Other Good Things to Know

- Binaries and libraries generated with ifort can be linked with binaries and libraries built with ifx, and .mod files generated with one compiler can be used by the other (64-bit targets only).
  - EXCEPT when built with `-ipo`

- Profile Guided Optimization (PGO)
  - Two flavors
    - Sampling: `-fprofile-sample-generate` and `-fprofile-sample-use`
    - Instrumented: `-fprofile-instr-generate` and `-fprofile-instr-use`
  - More information
    - https://clang.llvm.org/docs/UsersManual.html#profile-guided-optimization

More Good Things to Know

- `-check uninit`
  - New in ifx 2023.2.0 for Linux
  - Included in `-check all`
  - Compile
    - Add `-g` to get the symbols
    - Compile the whole application for best accuracy
  - MSAN == LLVM Memory Sanitizer
  - Set environment variable and then run
    - `MSAN_SYMBOLIZER_PATH=${ONEAPI_ROOT}/compiler/latest/linux/bin\llvm/llvm-symbolizer`

More Good Things to Know

- ifx only creates 64-bit binaries
  - `-m32` is not accepted

Known Issues

- COMPLEX data type performance
  - ifx 2023.x.x and older poor performance for COMPLEX data types
    - COMPLEX(KIND=4) same as Intel's COMPLEX*8
      COMPLEX(KIND=8) same as Intel's COMPLEX*16 or DOUBLE COMPLEX
      COMPLEX(KIND=16) same as Intel's COMPLEX*32
  - Improved performance in a future release

Creating
World
Changing
Technologies

Intel Fortran Compiler 2024

Sneak Peek

intel

Coming in 2024.0.0 – AddressSanitizer

- AddressSanitizer is a fast memory error detector.
- Detects:
  - Out-of-bounds accesses to heap, stack and globals
  - Use-after-deallocate
  - Use-after-return (to some extent)
- Will be available on Linux and Windows
- Typical slowdown introduced by AddressSanitizer is 2x.

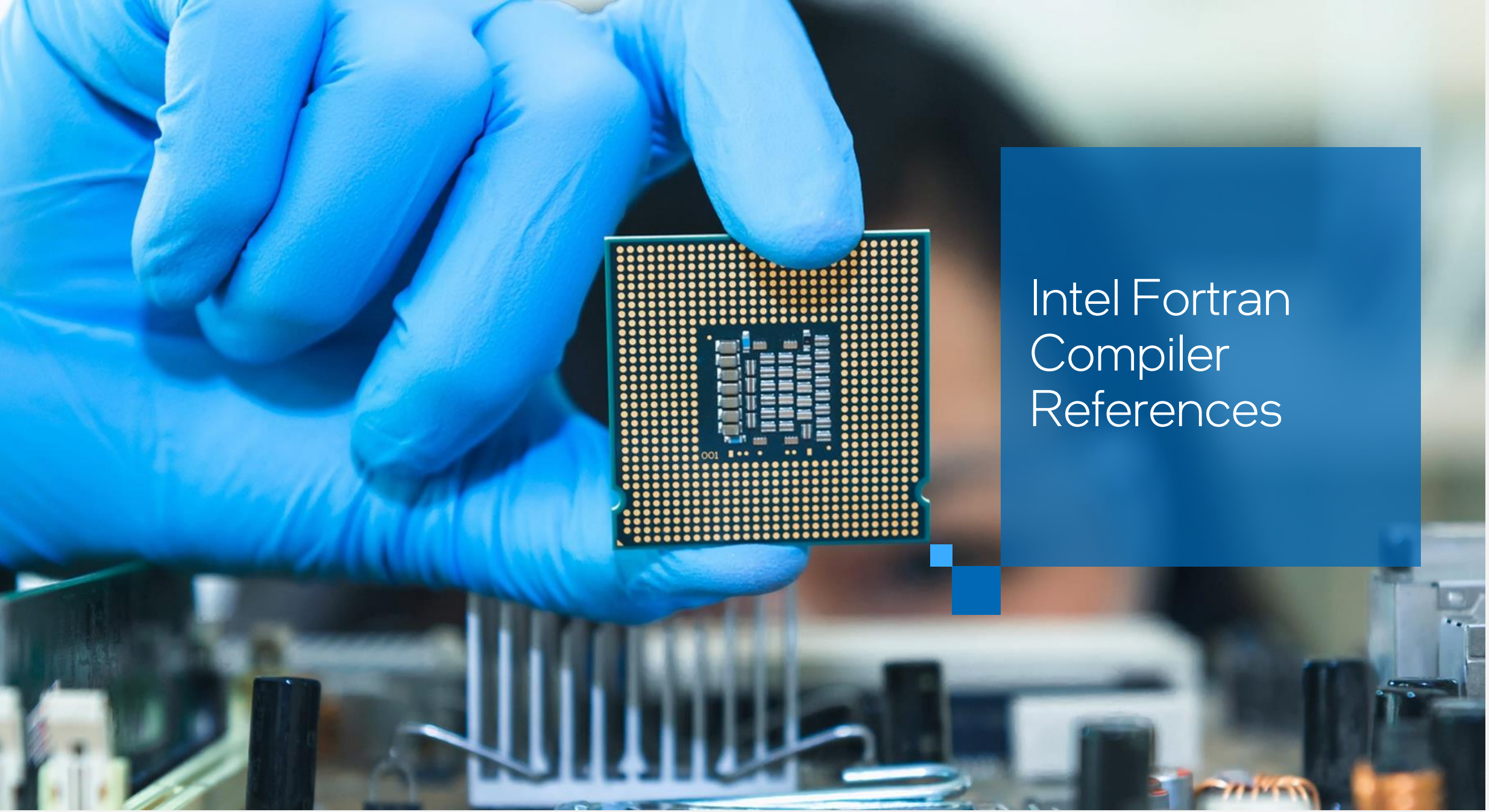
```
-fsanitize=address
```

https://clang.llvm.org/docs/AddressSanitizer.html

Also Coming in 2024.0 – ThreadSanitizer

- Detects data races in OMP and threaded code
- `-fsanitize=thread`
- Typical slowdown introduced by ThreadSanitizer is about **5x-15x**
- Typical memory overhead introduced by ThreadSanitizer is about **5x-10x**
- Linux only

https://clang.llvm.org/docs/ThreadSanitizer.html



Intel Fortran Compiler References

Intel Fortran Compiler Feature Support

Intel Fortran Language &
OpenMP Features Support
https://www.intel.com/content/www/us/en/developer/articles/technical/fortran-language-and-openmp-features-in-ifx.html

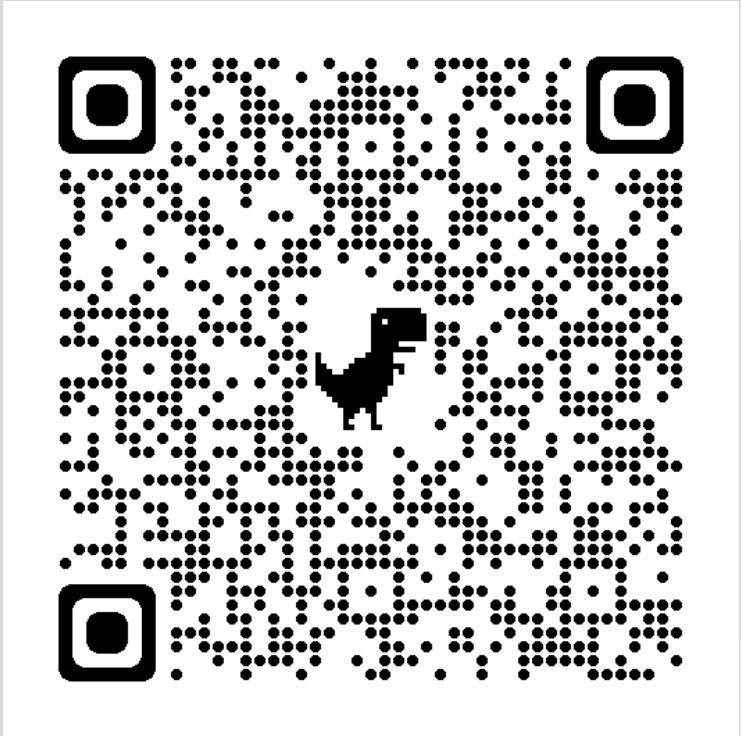
Porting Guide, ifort to ifx

Kept up to date with tips and techniques to help you move from ifort to ifx

https://www.intel.com/content/www/us/en/developer/articles/guide/porting-guide-for-ifort-to-ifx.html

Intel® Fortran Compiler Classic and Intel® Fortran Compiler Developer Guide and Reference

Fortran Developer Guide and Reference



OpenMP* Offload Basics

Learn the fundamentals of using OpenMP* offload directives to target GPUs through hands-on practice in this guided learning path

OpenMP* Offload Basics in DevCloud (with lab!)

See also this guide using matrix multiply on Github with the oneAPI-samples

Release Notes, System Requirements and more

- Release Notes

  https://www.intel.com/content/www/us/en/developer/articles/release-notes/oneapi-fortran-compiler-release-notes.html

- System Requirements

  https://software.intel.com/content/www/us/en/develop/articles/intel-oneapi-base-toolkit-system-requirements.html

- Driver downloads and installation guides

  https://dgpu-docs.intel.com/installation-guides/index.html

- Installation guides

  https://software.intel.com/content/www/us/en/develop/articles/installation-guide-for-intel-oneapi-toolkits.html


Questions?
Thank You for Attending!


intel

Notice & Disclaimers

- Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.
- Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.
- Your costs and results may vary.
- Intel technologies may require enabled hardware, software or service activation.
- © Intel Corporation. Intel, the Intel logo, and other Intel marks are trademarks of Intel Corporation or its subsidiaries. Other names and brands may be claimed as the property of others.