NASA and The Future of Fortran

Tom Clune*

NASA GSFC

*Material derived from Supercomputing 201
Steve Lionel – Intel Corporation

Applied Modeling & Simulation (AMS) Seminar Series
NASA Ames Research Center, April 28, 2015

Agenda

- Recent changes to the standard (F2003, F2008)
- Upcoming changes (F2015)
- Fortran standards process
- NASA engagement

The Present of Fortran

Fortran 2008 (ISO/IEC 1539-1:2010)

- Major features: coarrays, DO CONCURRENT, BLOCK, submodules
- Full implementation by Cray

Fortran 2003 (ISO/IEC 1539-1:2004)

- Major features: polymorphism and OO, C interoperability, IEEE floating control
- Full implementation by Cray, IBM, Intel, PGI

"Compiler Support for the Fortran 2003 and 2008 Standards"
updated 3X/year in ACM SIGPlan Fortran Forum:

http://portal.acm.org/sigplan/newsletter/fortran/

Fortran 2015 New Features

Integration of two Technical Specification documents

- TS29113 – Further C interoperability
- TS18508 – Additional parallel features

Limited new features and “wart removal”

- Intrinsics `COSHAPE`, `OUT_OF_RANGE`, `RANDOM_INIT`, `RANK`, `REDUCE`
- Procedures are recursive by default (unless `NON_RECURSIVE`)
- `IMPLICIT NONE (EXTERNAL)` disables implicit interface
- `EX` edit descriptor for hexadecimal significand
- `G0.`*d* can now be used for integer, logical, character

Further C Interoperability

- Guided by requests from MPI Forum
- Assumed type dummy arguments: `TYPE(*)`
- Assumed rank dummy arguments: `DIMENSION(..)`
- "C Descriptors" for passing more information to and from C
  - `ISO_Fortran_binding.h` declares C descriptors and procedures for manipulating them
  - Fortran compiler automatically creates/interprets these – no visibility from Fortran code
- Relaxed restrictions on dummy arguments to `BIND(C)` procedures:
  - `OPTIONAL`, `ALLOCATABLE`, `POINTER`, assumed-shape, deferred-length character now allowed

Additional Parallel Features

- Teams of images
- Events
- Atomic procedures (`ATOMIC_ADD`, etc.)
- Collective subroutines (`CO_BROADCAST`, `CO_MAX`, etc.)
- Ability to detect “failed images”

Newly Deleted and Obsolescent Features

Deleted features

- Nonblock `DO` (not ending on `END DO` or `CONTINUE`)

Obsolescent features

- Labeled `DO` (`DO 10 …`)
- Arithmetic `IF`
- `BLOCK DATA`
- `EQUIVALENCE`
- `COMMON`
- `FORALL`

Current Fortran 2015 Schedule

- 2015 – Technical work complete
- 2016 – Draft standard out for ballot
- 2017 – Publication

But… Parallel features TS not yet final – may affect schedule!

The Fortran Standard Committees

INCITS PL22.3 (aka “J3”)

- The U.S. Fortran Standards committee
- ~12 member organizations (mostly vendors and federally funded orgs)
- Meets 3 times per year
- http://j3-fortran.org

ISO/IEC JTC1/SC22/WG5

- the international Fortran Working Group
- determines the broad technical content of the Standards
- Delegates to PL22.3 as the Primary Development Body
- http://www.nag.co.uk/sc22wg5/

Changing the standard

Anyone can submit a proposed change, but …

- Helps to have an advocate at the meeting.

Anyone can attend meetings, but …

- Only members can formally vote.
- At most one member per organization, but multiple alternates

General advice:

- The committee prefers *use cases* over feature requests
- Preserve backward compatibility
- Consider implementation difficulty for vendors

NASA participation

I recently joined J3 as the *NASA* representative

- Note that JPL (Cal Tech) also has a representative

Need mechanisms to solicit requirements and priorities across the agency

How to communicate?

- Mailing list for NASA-centric discussions of the Fortran standard
  - fortran-users@lists.nasa.gov - low bandwidth
  - Requests/discussions for new features
  - Occasional polls for priorities (not a democracy)
- Or directly Thomas.L.Clune@nasa.gov
- There may be an alternate member at Ames (Henry Jin)

Potential priorities for Fortran 2020

- Improved facilities for generic programming
  - C++ Standard Template Library (STL)
  - Parameterized modules
- Improved testability of Fortran code
  - Program by contract
    - Mechanism to check preconditions, post-conditions and invariants on interfaces
  - Exception handling
- Checking physical units
- Extend OO – e.g. Java style interfaces
- Deferred rank arrays
- …

Thank you!

Questions / discussion