Fortran 2018 …and beyond

Steve Lionel, Convenor, ISO/IEC WG5 Fortran Standards Committee

August 2019

How is a new Fortran standard made?

- International Fortran committee is ISO/IEC JTC1/SC22/WG5
- Experts from individual countries make up National Bodies
- WG5 determines general content of the standard
- Development of features is done by the US National Body
- All WG5 members vote to approve a “Final Committee Draft”
- Fortran 2018 was published November 2018

Fortran 2018

TS18508 “Additional Parallel Features in Fortran”

- Technical Specification (TS) published in 2015
- Major new coarray features
  - Teams
  - Events
  - Failed Images
- Please refer to John Reid’s presentation for details

TS29113 “Further Interoperability of Fortran with C”

- Expands on C interoperability features first appearing in Fortran 2003
- Technical Specification published in 2012
- Designed in cooperation with the MPI Forum for Fortran interfaces to MPI

TS29113 continued

- In Fortran 2003 and 2008, these kinds of dummy arguments were not interoperable:
  - Assumed-shape arrays
  - Assumed-size arrays
  - Character length other than 1
  - Allocatable or pointer variables
- In Fortran 2018, all of these are now interoperable when a “C Descriptor” is passed

C Descriptor

- A C descriptor includes:
  - Attribute code (POINTER, ALLOCATABLE, OTHER)
  - Data type
  - Base address
  - Element length
  - Rank
  - Bounds and extents

C Descriptors

- Fortran creates and passes C descriptors to routines declared as BIND(C)
- C code can operate on C descriptors with CFI_xxx functions
- C code can create C descriptors and pass to Fortran
  - Fortran procedure must have BIND(C) attribute
- Descriptor layout, constants, functions declared in ISO_Fortran_binding.h

```
#include "ISO_Fortran_binding.h"
#include <memory.h>
#include <stdio.h>

extern "C" void greetings(CFI_cdesc_t * descr);

int main()
{
        int status;
        CFI_CDESC_T(0) cdesc;

        // Create our own local descriptor for an allocatable string
        status = CFI_establish((CFI_cdesc_t *)&cdesc, NULL,
                               CFI_attribute_allocatable,
                               CFI_type_char, 1, 0, NULL);
        //Allocate the string to length 7
        status = CFI_allocate((CFI_cdesc_t *)&cdesc, NULL, NULL, 7);
        // Copy in 'Hello, '
        memcpy(cdesc.base_addr, "Hello, ", 7);
        // Call Fortran to append to the string and print it
        greetings((CFI_cdesc_t *)&cdesc);
        printf("Length is now %zd¥n", cdesc.elem_len);
        status = CFI_deallocate((CFI_cdesc_t *)&cdesc);
}
```

```
subroutine greetings (string) bind(C)
    implicit none
    character(:), allocatable :: string

    string = string // 'Tokyo!'
    print *, string
    end subroutine greetings
```

```
Hello, Tokyo!
Length is now 13
```

Assumed Type

- Syntax is `TYPE(*)`
- Unlimited polymorphic - has no declared type
- May be used only for dummy arguments
- Like C `void`
- Limited use in Fortran code

Allocatable Dummy Arguments

- `ALLOCATABLE, INTENT(OUT)` dummy arguments get deallocated on entry to a Fortran procedure
- In Fortran 2018, a `BIND(C)` procedure can now have such an argument
- Fortran processor is required to do the deallocation on the call

More Interoperability Changes

- A Fortran procedure with a `CONTIGUOUS` dummy argument must be able to handle a C descriptor for a non-contiguous array
- Interoperable procedures may now have `OPTIONAL` dummy arguments
- `ASYNCHRONOUS` attribute extended to data access other than input/output

Assumed Rank

- Syntax is `DIMENSION(..)`
- May be used only for dummy arguments
- New `SELECT RANK` construct for use in Fortran code
  - `RANK(n)`
  - `RANK(*)` for assumed-size array
  - `RANK DEFAULT`

IEEE Floating Point Changes

- Many small changes to support IEEE 60559:2011
- “Denormal” is now “Subnormal”
- IEEE_GET_MODES and IEEE_SET_MODES gets and stores all the floating-point modes
- More rounding modes; separate modes for base 2 and base 10
- IEEE_INT, IEEE_REAL rounded conversion functions
- Quiet and Signaling comparison functions

IMPLICIT NONE enhancement

- `IMPLICIT NONE (EXTERNAL)` requires explicit interface or `EXTERNAL` declaration
- `IMPLICIT NONE (TYPE)` same as previous `IMPLICIT NONE`
- You can combine these:
  `IMPLICIT NONE (EXTERNAL, TYPE)`

More Changes

- Restrictions on constant expressions relaxed
  - `integer :: b = bit_size(b)`
  - `integer :: iota(10) = [ ( i, i = 1, size(iota,1) ) ]`
- `D0.d`, `E0.d`, `ES0.d`, `EN0.d`, `G0.d` and `Ew.dE0` edit descriptors
- `G0.d` may be used for integer or character values
- `STOP` and `ERROR STOP` can have any scalar expression
  - Optional `QUIET=` can disable messages

More Changes continued

- `ERRMSG=` added to `GET_COMMAND`, `GET_COMMAND_ARGUMENT`, `GET_ENVIRONMENT_VARIABLE`
- `OUT_OF_RANGE` intrinsic tests conversion
- `REDUCE` intrinsic applies user function to an array
- `COSHAPE` intrinsic like `SHAPE` for coarrays
- `RANDOM_INIT` specifies behavior of random number generator
  - Is sequence repeatable?
  - Does each image have its own sequence?

More Changes continued

- Arguments to `SIGN` intrinsic may have different kinds
- Non-polymorphic pointer arguments to `EXTENDS_TYPE_OF` and `SAME_TYPE_OF` need not be defined
- Kind of DO variable in implied DO may now be specified in array constructors and `DATA` statements
  - `[ (a(i,i), integer(long) :: i=1,n) ]`
- All procedures are `RECURSIVE` by default. New `NON_RECURSIVE` keyword
- Hexadecimal input and output with EX edit descriptor
  - Example: -15.625 with `EX14.4E3` might give `-0X1.F400P+003`

Deleted and Obsolescent Features

- Deleted
  - Arithmetic `IF` statement
  - Non-block `DO` construct
- Obsolescent
  - `COMMON` and `EQUIVALENCE`
  - Labelled DO statements
  - Specific names for standard intrinsics (for example `DACOS`)
  - `FORALL`

Future Revisions

- Next revision is informally called Fortran 202X
- Goal is to have it published no later than 2023
- Six-month survey of users 2017-2018
  - Results in WG5 document N2147

More Information

- WG5 web site https://wg5-fortran.org
  - Documents > N2161 The New Features of Fortran 2018
  - Fortran Standards > Fortran 2018

Questions?