Compiler Support for the Fortran 2003 and 2008 Standards

Ian D Chivers & Jane Sleightholme
Ian Chivers: Rhymney Consulting, London.
Jane Sleightholme: FortranPlus, London.
ian.chivers@chiversandbryan.co.uk
jane@fortranplus.co.uk

Introduction

This is a repeating article in Fortran Forum. The first version appeared in Fortran Forum in April 2007. The basis for the entries in the list of features was a report by John Reid. An electronic version can be found at:

`ftp://ftp.nag.co.uk/sc22wg5/N1601-N1650/N1648.pdf`

If you are a compiler vendor and would like to be included in future versions of this table please email one of us with details and they will be added to the table and published in Fortran Forum.

Acknowledgements for the original article

An email was sent to the J3 list asking for information about compiler support for the new features of the Fortran 2003 standard. The following people have contributed to the original article:

- Bill Long, Cray
- Joost VandeVondele
- Van Snyder
- Tobias Burnus and Brooks Moses, gfortran
- Andy Vaught, g95
- Robert Holmes, NAG

Thanks.

Revision 1

Two new compiler vendors were added. The information on the Intel compiler has been taken from the release notes that came with release 10 of the compiler. The information on the IBM entry has been taken from their web site. Ian Bush posted an article to comp.lang.fortran regarding this release (IBM XL Fortran Enterprise Edition for AIX, V11.1). Thanks Ian. See

`http://publib.boulder.ibm.com/infocenter/comphelp/v9v111/index.jsp?topic=/com.ibm.xlf111.aix.doc/getstart/new_features.htm`

for more information.

Intel and IBM were contacted to ask them to verify the information.

- Jim Xia of IBM corrected their entry, thanks Jim.
- Stan Whitlock of Intel corrected their entry, thanks Stan.

If there are any errors please notify us and we will correct them in the next version of this article.

Revision 2

Sun has been added and there have been a few corrections and updates to some of the other entries.

- Michael Ingrassia of Sun corrected their entry, thanks Michael.

Revision 3

The entries for Cray, gfortran (11 changes) , Intel (18 changes) and NAG (9 changes) have been updated.

Revision 4

We've had replies from Cray (Bill Long) , gfortran (Tobias Burnus), g95 (Andy Vaught), Intel (Stan Whitlock), Nag (Malcolm Cohen) and Sun (Robert Corbett).

We've also added two entries suggested by Richard Maine. Here is the text of the message we received from Richard.

> I just got the latest Fortran Forum and noticed two somewhat related Fortran 2003 features that I personally think are important, but aren't reflected in your table of features. If convenient, they might be useful to add to the table.
>
> 1. Allocatable scalars. To me, this is an important feature for object orientation, and in particular for polymorphism. Basically, a polymorphic object has to be either a pointer or an allocatable (or a dummy argument, which is a bit restrictive). In my experiments with polymorphism, the polymorphic objects pretty much always naturally "wanted" to be allocatable. But the NAG compiler (which I was using at the time) didn't yet support allocatable scalars. This meant that I either needed to make all the polymorphic objects pointers or make them arrays (possibly of size 1). Neither of these alternatives was attractive at all. I found this a significant enough shortcoming to keep me from using the polymorphic features. Thus, I'd think this would be something people would want to know about a compiler if they planned to use polymorphism.
>
> 2. Allocatable character length. I think that allocatable character length is one of the biggest "sleeper" features of f2003. It wasn't even on the list of f2003 requirements, and thus sometimes doesn't show up in lists of new features. It just naturally arise from allocatable length parameters for parameterized derived types. It seemed like one should allow the same thing for the one intrinsic type with a length type parameter. And lo, when it was all put together, it seemed like this was finally a good way to do variable length strings in Fortran. It integrates with the rest of the language immensely better than iso_varying_string has any hope of doing. In fact, as I said, it integrates so well that it came about as a consequence of the integration of other features. Allocatable-length character strings act like so many people intuitively think of character strings, unlike the fixed-length character strings that we've had since f77.
>
> Although this is related to allocatable scalars, in that you certainly want to be able to have allocatable character strings that are scalar, it is also a separate feature in that you can have allocatable scalars without necessarily allowing character length to be allocatable. It is also different in application, in that I see the main other usage of allocatable scalars as being for polymorphism, whereas allocatable character strings are not much related to polymorphism. It is also useful independent of parameterized derived types. I personally expect to see allocatable character strings used far more than parameterized derived types, even though it was the requirement for parameterized derived types that lead to allocatable character strings. I could almost see allocatable character strings as becoming the "normal" way that most character string variables are done.

Thanks Richard.

We've also added entries for the Fortran 2008 standard.

The last change are entries for compilers that support the Fortran 95 standard, and a list of compilers that are no longer under development but did support Fortran 90, and finally compilers which are available but we have no information on.

We have included the above for completeness. Given the widely differing levels of compiler standard conformance today we wanted to make this information available to people choosing a compiler.

Thanks to everyone who has provided the data.

Fortran 2003 and 2008 Features and Compiler Support: Revision 4

Y = Yes, N = No, P = Partial, U = Unconfirmed

Fortran 2003 features	Cray	gfortran	g95	IBM	Intel	NAG	Sun
ISO TR 15580 IEEE Arithmetic	Y	N	P	Y	Y	Y	Y
ISO TR 15581 Allocatable Enhancements	Y	Y	Y	Y	Y	Y	Y
Data enhancements and object orientation	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Parameterized derived types	Y	N	N	N	N	N	N
Procedure pointers	Y	P	Y	Y	Y	Y	N
Finalization	Y	N	N	Y	N	N	N
Procedures bound by name to a type	Y	P	N	Y	N	Y	N
The PASS attribute	Y	P	N	Y	Y	Y	N
Procedures bound to a type as operators	Y	N	N	Y	N	Y	N
Type extension	Y	Y	N	Y	Y	Y	N
Overriding a type-bound procedure	Y	Y	N	Y	N	Y	N
Enumerations	Y	Y	Y	Y	Y	Y	N
ASSOCIATE construct	Y	N	N	Y	Y	Y	N
Polymorphic entities	Y	N	N	Y	Y	Y	N
SELECT TYPE construct	Y	N	N	Y	N	Y	N
Deferred bindings and abstract types	Y	Y	N	Y	N	Y	N
Allocatable scalars, 12	Y	N			Y	Y	
Allocatable character length, 12	Y	N			Y	Y	
Miscellaneous enhancements	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Structure constructors	Y	Y	Y	Y	Y	N	N
The allocate statement	Y	P	P	Y	Y	Y	N
Assignment to an allocatable array	Y, 2	N	N	Y	Y, 2	Y	N
Transferring an allocation	Y	Y	N	Y	Y, 18	Y	N
More control of access from a module	Y	Y	N	Y	Y	Y	Y
Renaming operators on the USE statement	Y	P	Y	Y	Y	Y	Y
Pointer assignment	Y	N	Y	Y	N	Y	N
Pointer INTENT	Y	Y	Y	Y	Y	Y	N
The VOLATILE attribute	Y	Y	Y	Y	Y	Y	Y
The IMPORT statement	Y	Y	Y	Y	Y	Y	Y
Intrinsic modules	Y	Y	Y	Y	Y	Y	Y
Access to the computing environment	Y	Y	Y	Y	Y	Y	Y
Support for international character sets	N	Y	Y	P	P, 19	P	N
Lengths of names and statements	Y	Y	?	Y	Y	Y	Y
Binary, octal and hex constants	Y	Y	Y	Y	Y	Y	Y
Array constructor syntax	Y	Y	Y	Y	Y	Y	N
Specification and initialization expressions	Y	P	Y	Y	P	P	N
Complex constants	Y	Y	Y	Y	Y	Y	Y
Changes to intrinsic functions	Y	P, 9	Y	Y	P	Y	N
Controlling IEEE underflow	Y	N	N	Y	Y	N	Y
Another IEEE class value	Y	N	N	Y	Y	N	Y
	Cray	gfortran	g95	IBM	Intel	NAG	Sun

Input/output enhancements	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Derived type input/output	Y	N	N	Y	N	N	N
Asynchronous input/output	Y	Y,10	Y	Y	Y	Y	Y
FLUSH statement	Y	Y	Y	Y	Y	Y	Y
IOMSG= specifier	Y	Y	Y	Y	Y	Y	Y
Stream access input/output	Y	Y	Y	Y	Y	Y	Y
ROUND= specifier	Y	N	P	Y	Y, 20	Y	Y
DECIMAL= specifier	Y	Y	Y	Y	Y, 21	Y	Y
SIGN= specifier	Y	Y	Y	Y	Y, 22	Y	Y
Kind type parameters of integer specifiers	Y	N	?	Y	Y	Y	N
Recursive input/output	Y	P	Y	Y	Y	Y	Y
Intrinsic function for newline character	Y	Y	Y	Y	Y	Y	N
Input and output of IEEE exceptional values	Y	Y	Y	Y	Y	Y	Y
Comma after a P edit descriptor	Y	Y	Y	Y	Y	Y	Y
Interoperability with C	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Interoperability of intrinsic types	Y	Y	Y	Y	Y	Y	Y
Interoperability with C pointers	Y	Y	Y	Y	Y	Y	Y
Interoperability of derived types	Y	Y	Y	Y	Y	Y	Y
Interoperability of variables	Y	Y	Y	Y	Y	Y	Y
Interoperability of procedures	Y	Y	Y	Y	Y	Y	Y
Interoperability of global data	Y	Y	Y	Y	Y	Y	Y
	Cray	gfortran	g95	IBM	Intel	NAG	Sun

Notes

1	No unlimited polymorphics
2	Optional under flag
9	kind= of maxloc, minloc, shape missing
10	implemented as synchronous i/o
12	Suggested by Richard Maine
18	MOVE_ALLOC
19	SELECTED_CHAR_KIND only
20	plus RC,RD,RN,RP,RU,RZ
21	plus BLANK=,DELIM=,PAD=,SIZE=
22	plus DC,DP

Fortran 2003 and 2008 Features and Compiler Support: Revision 4

Y = Yes, N = No, P = Partial, U = Unconfirmed

Fortran 2008 Features	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Submodules	Y	N			N	N	N
Coarrays	P	N	P		N	N	N
Performance enhancements						N	N
do concurrent	N	N			N	N	N
Contiguous attribute	N	N			N	N	N
Simply contiguous arrays	N	N			N	N	N
Data enhancements						N	N
Maximum rank	N	N			N	N	N
Long integers	Y	Y			Y, 100	Y	Y
Allocatable components	N	N			N	N	N
Implied-shape array	N	N			N	N	N
Pointer initialization	N	N			N	N	N
Kind of a forall index	N	N			N	N	N
Allocating a polymorphic variable	N	N			N	N	N
Accessing data objects	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Accessing real and imaginary parts	N	N			N	N	N
Pointer functions	N	N			N	N	N
Input/Output						N	N
Finding a unit when opening a file	N	Y			N	N	N
g0 edit descriptor	N	Y			N	N	N
Unlimited format item	N	N			N	N	N
Recursive input/output	Y	Y			Y	Y	Y
Execution control	Cray	gfortran	g95	IBM	Intel	NAG	Sun
The block construct	N	N			N	N	N
Exit statement	N	N			N	N	N
Stop code	N	N			P, 104	N	N
Intrinsic procedures for bit processsing						N	N
Bit sequence comparison	N	N			N	N	N
Combined shifting	Y	N			P	N	N
Counting bits	Y	P			Y	N	N
Masking bits	N	N			N	N	N
Shifting bits	Y	N			P	N	N
Merging bits	N	N			N	N	N
Bit transformational functions	N	N			N	N	N
Intrinsic procedures and modules	Cray	gfortran	g95	IBM	Intel	NAG	Sun
Storage size	N	N			N	N	N
Selecting a real kind	N	N			N	N	N
Hyperbolic intrinsic functions	N	P			Y	N	N
Bessel functions	N	P			N	N	N
Arc tangent function	N	P			N	N	N
Error and gamma functions	N	P			P	N	N
Euclidean vector norms	N	N			N	N	N

Parity	N	N			N	N	N
Execute command line	N	N			N	N	N
Location of maximum or minimum value in an array	N	N			N	N	N
Find location in an array	N	N			N	N	N
Constants	N	N			N	N	N
Module procedures	N	N			N	N	N
Programs and procedures						N	N
Empty contains section	Y	Y			N	N	N
Internal procedure as an actual argument	N	N			Y	N	N
Generic resolution by pointer or allocatable attribute	N	N			N	N	N
Null pointer as a missing dummy argument	Y	N			N	N	N
Elemental procedures that are not pure	N	N			N	N	N
Entry statement becomes obsolescent	N	N			N	N	N
	Cray	gfortran	g95	IBM	Intel	NAG	Sun

Notes

100 INTEGER (KIND=8)

104 not STOP <init expr>

Fortran 95 and TR 15581

The following companies also make Fortran compilers and we include details of their degree of standard support.

	Absoft	**Salford ftn95**	**Lahey**	**Path Scale**	**PGI**
Windows					
Fortran 95	Yes	Yes	Yes		Yes
TR 15581	No	No	Yes		Yes
Linux					
Fortran 95	Yes		Yes	Yes	Yes
TR 15581	No		Yes	Yes	Yes
Others					
Apogee		No longer available			
Compaq		No longer available			
Fortran Company		F			
Fujitsu		No information at this time			
Hewlett Packard		No information at this time			
NA Software		No longer available			
NEC		No information at this time			