Fortran 8x: the Canadian position: August 1987

1.0 Introduction

For several years. the Canadian Fortran group has prepared a paper on its position on the proposed Fortran 8x standard for the annual WGS meeting. This current paper updates our position, and re-states some of our previous observations.

2.0 Comments on the results of the WG5 and X3J3 letter ballots

While we voted 'yes' in the informal WGS letter ballot. like several other WG5/X3J3 members this did not represent the fact that we were satisfied with the draft proposed standard as it stood but that we felt it was time that the 8x document be made available to a wider audience for comment. Therefore, our comments were primarily editorial, to aid the production of a coherent, readable document for public review. It remains our hope that the public comment period will enable the 8x document to converge more quickly to an acceptable Fortran standard.

We are pleased to note that the thorough processing of the letter ballot comments by X3J3 has resulted in a much-improved document S8.104. However. the letter ballots themselves raised a number of concerns for us:

1. There is still disagreement on the fundamental approach to the standardisation process (e.g. Harris, Lakhwara, Weaver), and we expect that this will not be resolved.

2. It is disconcerting that some of the major vendors are still strongly opposed to Fortran 8x.

3. Despite our ongoing concern with the size of the language. we have some sympathy for the re-introduction of the bit data type, as requested in several of the ballots (e.g. Metcalf, Moss) since the introduction of an additional data type does not change the nature of the language.

4. The size and nature of the language is still causing concern. Although Fortran 8x is 'only' about 50% larger than Fortran 77, measuring in terms of keywords or syntax rules, the nature of Fortran 8x is significantly different from Fortran 77. While many of the traditional Fortran features are retained (e.g. COMMON, EQUIVALENCE), they are no longer needed in 8x (due to the introduction of new features), and their use is not recommended. These obsolescent and deprecated features constitute almost 25% of Fortran 77. Thus only about half of Fortran 8x consists of surviving Fortran 77 features.

3.0 Requirements of the Fortran language standard

We feel that the Fortran language standard should define a language that is easy to learn and use. that is suitable for both the casual scientific/engineering user and the experienced" professional programmer. While each new feature of Fortran 8x has its individual merit, we remain unconvinced that all the new features proposed in Fortran 8x will be of value to a substantial fraction of the user community. We are concerned that they will contribute adversely to the cost and the quality of implementation, for both compilation and execution.

We are also concerned that the new Fortran standard is overdue. Much of this problem has been due to the ambitious attempt to introduce many contentious new features into the language. Despite the effort and high quality of the work done by X3J3, consensus is difficult to reach when new ground is being broken.

4.0 Classification of Fortran 8x language features

The table on the next page is an attempt to classify some of the main features of Fortran 8x and some current Fortran implementations, in order to be able to consider the merits of each in the context of the whole language. The list also includes some features which have been recently moved to the ‘Removed Extensions’ appendix, but upon which we would like to comment.

This classification arises from the guidelines that we feel can be applied to the production of a new standard for an existing language:

- remove unused or obsolete features of the old standard,
- standardise common extensions of the old standard,
- adopt common features from other languages,
- include new features.

5.0 Obsolescent and. deprecated features

We support the list of obsolescent features in Fortran 8x, since it is essential to have a mechanism to allow ancient features to be deleted from the language.

The list of deprecated features is also reasonable (assuming that the new features which replace them remain in Fortran 8x), since they will be present in the next revision of the standard, at least. However, we note that there is no list of deprecated features actually in the proposed standard; it is only specified in Appendix B, which "is not part of ANS X3.9-l98x, but is included for information only".

Obsolescent and deprecated features	**Common extensions to Fortran 77**
Obsolescent features: • arithmetic if • real do variables • shared do terminal statement • branch to endif from outside its if block • alternate return • pause • assign and assigned go to • assigned format specifiers Deprecated features: • storage association • passing array element substring to dummy array • block data • entry • common, dimension, equivalence • fixed form source • specific names for intrinsic functions • statement functions • computed go to statement • list-oriented data statement • double precision statement • **char_length* in type specifier	Features included in Fortran 8x: • namelist • ! comments • free-form source format • lower-case names • long names • use of underscore in names • alternate form of relational operators (e.g. = =) • implicit none • " to delimit character constants • enddo • lengths of real variables • random, clock and date intrinsics • recursive procedures Features not included in Fortran 8x: • include • hexadecimal constants • octal constants • binary constants • lengths of integer and logical variables • do while • bit manipulation intrinsics
Features common in other languages	**New features, not common in other languages**
Features included in Fortran 8x: • ; as statement separator (multi-statement lines) • structures (records) • whole array operations • allocatable arrays • case construct • exit and cycle • internal procedures • intent specification • inquiry intrinsics • character intrinsics Features not included in Fortran 8x: • significant blank • bit data type • simple pointers • allocatable structures • variant structures	Features included in Fortran 8x: • real(*,*) • modules and use • parameterized user types • user-defined operators • array sections • range • identify • alias • where • optional and keyword dummy arguments • many array intrinsics Features not included in Fortran 8x: • vector-valued subscripts • forall • exception handling

Figure 1. Classification of Fortran 8x features

6.0 Common extensions. to Fortran 77

In general, we support the inclusion in Fortran 8x of those extensions to Fortran 77 which are now commonly available. In particular, the simpler extensions to the lexical elements of the language (e.g. free-form source, ! comments, lower case, long names) are 'obviously desirable'. We have comments below on some of these features which are not currently included in Fortran 8x, but most of which we expect vendors to continue to support. If vendors are likely to continue to support these features, is this good reason for standardising them ?

6.1 Lengths of integers and logicals

We expect vendors will continue to offer short integer and short logical data types(e.g. integer*2, logical*l), even if they are not included in Fortran 8x. They are not particularly portable and they are not aesthetic, but they do get the job done.

6.2 Hexadecimal, octal and binary constants

Although the hexadecimal representations of characters is available through the CHAR intrinsic function, we see no good reason why these very common extensions should not be standardised.

6.3 Include

The include is a commonly provided extension, which is simple to understand and easy to implement. Even if it is not included in Fortran 8x, vendors are still likely to support it in the long term. However, if modules remain in Fortran 8x, then we agree that include is redundant.

6.4 Namelist

We were satisfied with the standardisation of namelist made in Fortran 8x since the 1986 letter ballot.

6.5 Do while

Although the do while construct is a common extension to Fortran 77, we see no reason for its inclusion, given the do constructs provided in Fortran 8x.

6.6 Bit manipulation intrinsics

Since the bit data type is not currently included in Fortran 8x, we are satisfied with the availability of these intrinsics in the supplementary standards for specialised users (as specified in S8, appendix A).

7.0 Features common in other languages

Some of the proposed Fortran 8x features are new to Fortran. yet are commonly found in other languages. Since such features are generally well-understood by both compiler writers and language users, in general we support their presence in Fortran 8x. We also have some comments on those features which are not included.

7.1 Structures

A structure facility is clearly required in a new Fortran standard. and we support its presence in Fortran 8x, via derived data types.

7.2 Case,.exit and cycle

These statements have proved useful in other languages, and are they simple to implement and understand. We support their inclusion in Fortran 8x.

7.3 Internal procedures

We feel that internal procedures should essentially be multi-statement statement functions, and we support the simplification that has been made in Fortran 8x.

7.4 Interface (intent) specification

We support this relatively simple addition to the language. It provides useful function at little cost.

7.5 Character and inquiry intrinsics

We support the inclusion of these intrinsics. since they promote the production of portable code.

7.6 Significant blank

We still support the inclusion of the significant blank in the new free-form source (while retaining the insignificant blank in the old fixed-form source) at this opportune time. We were not convinced with the response by X3J3 to our letter ballot that the insignificant blank 'is something that many long-time Fortran users like and want to retain'. This is not in agreement with the Halifax WG5 vote, which was heavily in favour (28-4) of the significant blank. We believe that most Fortran programmers write code which looks 'natural', with blanks interspersed between words to improve readability, and not imbedded in keywords and symbolic names to hinder readability (though we accept that there may be a small number of exceptions to this, such as imbedded blanks in long numeric constants). Also, if they prefer to write code with insignificant blanks, then they may still continue to do so in the fixed-form source.

7.7 Bit data type

As we stated in our comments on the letter ballot earlier, we would be happy for the inclusion of this feature to be reconsidered. We feel it is a feature which fits into the traditional nature and role of Fortran. We noted that this feature was requested by most of the 'no' votes in the letter ballot. Hopefully the public review period will provide guidance on the demand for this feature.

7.8 Pointers

We would like to see a simple pointer facility introduced in Fortran 8x, but we appreciate that it would now mean significant effort at this stage. Much of the underlying implementation for pointers will be necessary to support some of the features of Fortran 8x (e.g. allocate), yet pointers will not be explicitly available to .users. Further, if pointers are to be introduced into Fortran then it would be preferable that this is done before the new aliasing mechanisms, IDENTIFY and RANGE, have been incorporated into the standard. However, we are happy for the public review period to guide X3J3 on the user demand for this feature.

7.9 Allocatable structures

Although the vote on allocatable scalars failed (11-l5-6) at the Halifax WG5 meeting, we are concerned about the implication on structures (which are scalars of derived type, but could contain large arrays). We feel allocatable structures could be a reasonable requirement, though this introduces the inconsistency between scalars of intrinsic and derived types.

7.10 Variant structures

Variant structures are not currently included in Fortran 8x, and we would like them to remain absent.

8.0 New, unique or unusual features

These are the features which cause us most concern. We appreciate that every individual feature is useful to someone, and that it has gained sufficient support in X3J3 for inclusion in 8x, yet we doubt that they are useful to most of the Fortran community. If these features were already in g-eat demand, we feel that vendors would already have provided some implementation, but such is not the case.

8.1 real(*,*)

We feel that the combinatorial problems with real(*,*) dummy arguments, which existed in earlier versions of S8, have been adequately resolved (by limiting the number of arguments which can be passed with real(*,*)) in the current version of Fortran 8x.

8.2 Modules and use

While we appreciate the functionality provided by modules and use. we feel that these features have significant impact on the nature of the language and its implementation (despite the fact that it is only 3% of the language (ref. X3J3 response to Lakhwara)). We are concerned with the lack of practical experience with such features, in both implementation and use. The latest letter ballot and the responses from X3J3 did not fully alleviate our concerns about these features. and so we remain unconvinced that they should be included in the current revision of the Fortran standard.

8.3 Derived data types and user-defined operators

Although derived data types and user-defined operators provide a convenient structure facility, we see operator overloading as an unnecessary extension that should be excluded.

8.4 Identify and alias

Though we are concerned with the impact on the nature of the language, a simple aliasing mechanism, via identify, is acceptable to us.

8.5 Range

We would prefer to see range removed from Fortran 8x. since similar functionality can be obtained with identify and array sections.

8.6 Array extensions (forall and vector-valued subscripts)

These features have been moved to the ‘Removed Extensions’ appendix, and we would prefer that they are not re-introduced into the language.

8.7 Exceptional handling

We do not support the inclusion of exceptional handling in Fortran 8x. but we would be happy to see this feature defined in an appendix or journal of development.

9.0 Conclusion

We have been pleased that X3J3 has attempted to reduce the size of the proposed Fortran 8x standard. and some of our concerns have been partially addressed since the first letter ballot. However, the second letter ballot indicates that there is still much work to be done to achieve consensus.

We feel it is important that a new Fortran standard should be available in the near future, but we urge that it should indeed be a standard that will gain widespread acceptance and implementation for these are the criteria by which the standard will be measured.

Canadian Standards Association, Fortran Working Group