Minutes of the

Fortran Experts Group

Meeting at Geneva, 9-12 April 1984

X3J3/163

[This title page is not strictly correct. The Fortran Experts Group had been formally established as SC5/WG9 at the SC5 Plenary meeting in Ottawa in September 1983]

CONTENTS

Page

AGENDA

Geneva, Switzerland
April 9 -12, 1984

Monday, April 9, 1984, 9:30 A.M.

1. Opening of the meeting (Jeanne Martin, Convener)
2. Welcome of delegates
3. Roll call of delegates (Mario Surdi, Secretary)
4. Election of chair
5. Minutes of the Vienna Meeting (June 1982)
6. Adoption of agenda
7. National activity reports (Heads of Delegations)
8. Procedural matters
9. Restructuring of TC97 (Jeanne Adams, Chair, SC5)
10. Summary of X3J3 Actions since June 1982 (Jeanne Martin, Secretary, X3J3)

Monday, April 9, 2:00 PM

11. General Overview - Document X3J3/S7.89

(S7 is constantly undergoing revision. It currently consists of five major sections, and this breakdown seems to be fairly stable. The revisions occur within the sections. Our review is thus organized into five sessions that correspond to the five sections of the document. At the present time, Event Handling is covered in a separate chapter, so there is a separate session for this topic.)

	Discussion Leaders (Moderator)
I. Introductory Concepts	Marshall (Matheny)
a. General	Hirchert (Hendrickson)

Tuesday, April 10, 9:00 AM

b. Arrays	Paul (Marusak)
III. Statement Concepts	Matheny (Marusak)

Tuesday, April 10, 2:00 PM

IV. Event Handling	Koblitz (Muxworthy)
V. Unit Concepts	Crowley (Snoek)
VI. General Concepts	Wagener (Hendrickson)

Wednesday, April 11, 9:00 AM

12. Liaison Activities

Graphical Kernel System Fortran Binding — Wagener (Hirchert)
Reference Document: N 762 Fortran Interface of GKS 7.2

Wednesday, April 11, 2:00 PM

13. Tour of CERN (4:00 -6:30 PM)

Thursday, April 12, 9:00 AM

14. Comments from the Delegates

15. Presentations by the Delegates

    Results of the Questionnaire about the present use of the Fortran language and the desired orientation of the future standard Fortran 8X. Presenter Mas.

    Results from SEAS Questionnaire on the attitudes toward Fortran and directions for Fortran 8X. Presenter Metcalf.

    Standard Method of Specifying Requirements for Fortran language Processors. Presenter Meek.

Thursday, April 12, 2:00 PM

16. Remaining Comments and Presentations

17. Recommendations

18. Requirements concerning a subsequent meeting

19. Closing Business

20. Adjournment

ATTENDANCE LIST

1. Jeanne Adams, United States
2. Cornelius G.F.Ampt, The Netherlands
3. Bert Buckley, Canada
4. Paul Alan Clarke, United Kingdom
5. Ted Crowley, United States
6. Ingemar Dahlstrand, Sweden
7. Jeremy J. DuCroz, United Kingdom
8. Francoise Ficheux-Vapne, France
9. Richard Hendrickson, United States
10. Kurt W. Hirchert, United States
11. Werner Koblitz, Austria
12. Ulrich Kulisch, Germany
13. Alain Leteinturier, France
14. Herman Luttermann, Germany
15. Neldon H. Marshall, United States
16. Jeanne T. Martin, United States
17. Alex Marusak, United States
18. Christian Mas, France
19. James Matheny, United States
20. Brian L. Meek, United Kingdom
21. Michael Metcalf, Switzerland
22. David Muxworthy, United Kingdom

23. Ikuo Nakata, Japan

24. George Paul, United States

25. Aurelio Pollicini, Italy

26. Karl-Heinz Rotthaeuser, Germany

27. Gerard J. Schmitt, Austria

28. Mok-Kong Shen, Germany

29. Jan A.M. Snoek, The Netherlands

30. Hieronymus Sobiesiak, Germany

31. Mario Surdi, United States

32. Christian Ullrich, Germany

33. David M. Vallance, United Kingdom

34. Nico Vossenstijn, The Netherlands

35. Jerrold Wagener, United States

36. John D. Wilson, United Kingdom

SYNOPSIS OF THE MEETING

1. The meeting was opened at 9:30 AM by the Convener, Jeanne Martin

2. The delegates were welcomed by Prof. Ian Butterworth, Director of Research, CERN

3. Delegates were present from the following countries: Austria, Canada, France, Germany, Italy, Japan, Netherlands, Sweden, Switzerland, United Kingdom and United States.

4. Jeanne Martin was elected to chair the meeting; David Muxworthy was elected Vice-Chair.

5. Corrections were made to the Minutes of the Fortran Experts Group Meeting at Vienna June 14 - 17, 1982.

6. The agenda was amended and adopted (the adopted agenda appears on Page 1).

7. National activity reports were presented by the delegates from each of the countries.

   References: Al, A2, A3, A4, A5.

8. It is customary to take straw votes to record opinions. Everyone may vote (Yes-votes, No-votes, undecided).

9. Restructuring of TC97

   References: C-18, WG 9 Position Papers.

   STRAW VOTE: Would you like to see a resolution proposing that we disagree with the reorganization for the reasons put forth? (18, 0, 11)

10. Summary of X3J3 Actions

    References: B-1, X3J3 Formal Votes since 6/82.

II. GENERAL OVERVIEW -DOCUMENT X3J3/S7.89

I INTRODUCTORY CONCEPTS

References: B-2, Core-plus-modules Model

The presentation contrasted the old module concepts of "core plus modules" with the current single "core plus deprecated features" model of FORTRAN 8X. Emphasized were deprecated features, abstract data type => derived data type => bundle.

Discussion:

Ampt	This is a natural progression, but in '97 with PC's et al and new applications, only core Fortran will be around. Hence this is a mistake. We should treat core as a subset.
Paul	It is not a problem to implement on a PC
Ampt	Wrong way
Marusak	Is the concern complexity of a compiler, or of the language?
Ampt	The language. It's still a general language
Hirchert	There should be two levels of conformance. There is now a single language for processors. For teaching, there are no objections. Even for the PC, there is old code around.
Wilson	Why call it Fortran if deprecated features are omitted?
Snoek	Concerned with the document -- one document. Better if deprecated features were removable. The idea, what is obsolete, are now known.
Ampt	Assign is still there.
Wilson	There is a difference between the Standard and books. There are now two Fortran 77 Standards, but vendors say merely "Standard".
Wilson	Look at the sales literature.
Ampt	Don't talk about deprecated features. Arrange document.
Wilson	How many people learn from Standards document?

Ampt The standard is the ultimate text.

Dahlstrand The text may be pretty good re above discussion. It is very close to a core subset.

DuCroz I support Ingemar. A good text processor should be able to edit out deprecated features.

STRAW VOTE The text of the Standard should be such that deprecated features can be left out. (22-2-5).

Vallance Should we require that instances of use of deprecated features be flagged?

Hirchert The Standard has few words to say about requirements on processors.

Hendrickson Some non-standard things are not detectable at compile time, et al.

Paul The opinion is that deprecated features will definitely be removed. The next committee will do what it wants.

DuCroz Indeed? We can still mark them.

Meek This topic has received a good deal of thought. Mark them. Then you can get rid of them. I support F8X. But I would go further, would like to see requirements on a processor to accept only core. This is effectively a subset. A third option is the changing of code to something which is core-conforming.

Hirchert Marking is important. F9Z will be a superset of core.

Hendrickson Not all of the problems are syntax. Some things are difficult to detect.

STRAW VOTE A Fortran processor should be required to optionally flag the use of deprecated features (18-2-11).

Dahlstrand Can't do it within the text constraints of Standardese.

Schmitt The font of deprecated features is not clear, and it should be.

Meek I support this, we should use white ink.

Marusak Originally these were " obsolete" features. Should we say "deprecated"?

Hirchert There is a difference. Deprecated is not a module off in a corner. It must be part of the language.

Matheny Our experience with a flagger is that people don't use it.

Metcalf We require the Standard at CERN.

Hirchert More people are more and more aware.

Meek Re the /S7 document. You say that deprecated features are a part of the language. In fact they are a separate Section 18. They should be in an appendix. Re meta-language. It isn't, not a full language, with production rules. It ought to be. /S7 could get to be a proper meta-language.

Martin We are only on the way.

Hirchert We haven't gotten there yet.

Paul I agree with Brian, but not regarding appendix. This must be in the text. If too strange, they are going to remove the X3J3 committee.

Meek You are standardizing the reasons for deprecating.

Buckley I support the idea that deprecated features belong in an appendix.

Metcalf I support this. In Fortran 8X, section 18 belongs in the appendix. The information is excellent.

Marusak Re the FIB, questions, survey -- these are extremely useful, and tend to deal with the revision. Re the discussion of deprecated features -- is the list the current one? Nothing in F9Z can be removed unless deprecated in F8X.

Hendrickson Section 18 is the way it is for two kinds of reasons: COMMON must be deprecated to get new stuff; Arithmetic IF is just bad, religiously.

Meek Probably the list is right.

Metcalf Five people want to retain the arithmetic IF.

Clarke Of the list, four features are Fortran 77.

Discussion on difficulty of getting Fortran 77 accepted:

Metcalf	Can't introduce new standard without support from manufacturers, especially the larger ones. Slowness in getting Fortran 77 accepted lies squarely at the door of IBM
Muxworthy	The fault has not been totally with IBM. The main problem has been the reliability and efficiency of the compilers
Vallance	Another problem is that many user programs are written in F66 (e.g. CADCAM software)
Marusak	66 to 77 has many trivial changes. The change from Hollerith to character. When does one convert?
Ampt	There were more basic problems especially at the universities. More education, formal training is required. It is not just a vendor problem.
Meek	It is not just education. The production of a product must be sold through publicity (FORTEC, UK trade press, Computer Weekly). Also users must write into their contract a mandatory standard conforming compiler.
Ampt	The technical magazines do not have the right people for the promotion of FORTRAN.
Schmitt	In industry there was a low usage. When a company brought out a compiler they installed the current language. In the technical universities 80% FORTRAN 77, Graphic programs only in 66, new programs are still written in Fortran 66.
Snoek	IBM sites started with Release 2. 400 compilations/week - not sure how much is 66 . CDC change was more gradual. Users were satisfied with results.
Paul	In early 76 there was a movement to Fortran 77. Product shops were resistant to the change. There were no customer requirements for Fortran 77. When it became a FIPS standard the new compiler became a requirement. 77 Fortran didn't do anything for the scientific user. The universities said Fortran was dead. They didn't require students to use Fortran. The Engineering/ Scientific Community doesn't really care about modern language aspects. It's job oriented. Give them something they really want (eg array extensions).

Hirchert	CDC was abrupt in some aspects. The 77 Fortran object programs were not quite compatible with 66. Upgrading Fortran version 77 required version 66 to be upgraded. The Block IF was wanted but users don't want to be the first to use new compilers.
Schmitt	Technical universities were upset that no new computer supported Fortran 77.
Paul	Computer Science students in the United States must pass a proficiency test in PASCAL. Fortran is still the second most used language.
Meek	We must ensure that students live in the real world. There are language snobs in computer science departments. Purpose of a standard is to promote portability. Standard is driven by a minority of users. If we want to sell the standard it must service all users. Fortran 77 would be more popular if it had a bit data type. Part of the standard should include quality and performance.
Buckley	I agree with what George said. Fortran 77 was not accepted because it didn't include what the scientist wanted. Computer Science graduates have not seen Fortran. Fortran is not a dead language. If the next standard is well done we can n convert PASCAL users.
Lutterman	80% of the programs in the universities are written in Fortran. In 81 they switched to Fortran 77. There are no problems with the new generation of students. They only know Fortran 77. Problems exist with converting the previous generation of users.
Hirchert	100%. of computer scientists use PASCAL. Numerical analysis and engineers use Fortran.
Marusak	Do not understand the value of character type. At Los Alamos all work is done in Fortran.
Paul	70-75%. of all user work is done in COBOL. Less than 25%. in Fortran. Less than 5% all others.
DuCroz	It will be a serious problem if there are any incompatibilities between Fortran 77 and 8X. Reliability and efficiency are not questions regarding this standard.

II. Data Concepts

a.1. GENERAL

References: B-3, Data Concepts - General
Discussion:

Schmitt

My students have most problems understanding these chapters. Arrays vs. new arrays is poor. Need re-write. Chapter on numerical approximation seems to be missing a section on precision. Comment: Began to take action at 89 meeting, no time to do it. We agreed we need to “clean-up." Will try again. Should be careful in S7 about text, sometimes use MAX or max; > or .GT. Use standard Fortran.

Metcalf

1) If no varying length character we need a function that gives length to last non-blank. Comment: Canadian proposal coming which does several string things in forward and backward direction.

2) Physicists need pointer, they are willing to run risk of storage association problems.

3) They need efficient bit data type. Derived data won't be good enough. Should be easy to extract from S6, or at least, INTEGER*2 for bytes.

Buckley

1) Must have pointer, else derived data types are useless. Syntax isn't important.

2) Remove entity oriented declarations. Don't fit into current existing framework. Duplicate existing stuff.

3) Character function proposal coming.

4) Need more regular syntax in character, (e.g. A(I:)) to mean rest of string. Comment: it is in F-77.

Shen

If we have heap anyhow, why not allow varying upper bound?

Ampt

Problems with bit data type IRTF has it, but assumes 2's complement, etc. We need to get away from hardware. Possibly logical. Implementor could do it efficiently.

Do we want "strings of bit values." Then don't have SHIFT etc. since it is so hardware oriented.

We need conversion intrinsics from string of bits to integer.

Metcalf We have 160,000 tapes of bits. They are the fundamental data type and we must deal with them efficiently. This is the real world.

Ampt Don't you need I/O into strings of bits.

Metcalf We need to get efficient implementation.

a.2 POINTER DATA TYPE

At the next X3J3 meeting a proposal to introduce a POINTER data type will be made. Pointers can point to either derived data types or arrays. There will be strict type checking on pointers. Once declared a pointer cannot point to a different derived data type or array of different rank. The name of the pointed-to object is a normal Fortran name. The pointer name is the object name suffixed with a ¢. (The actual suffix will be determined at a later time).

Examples:

```
TYPE   PERSON
       AGE:   REAL
       NEXT GUY¢:   POINTER  (PERSON)
END TYPE

DAD¢:   POINTER (PERSON)

RALPH: TYPE (PERSON)
```

The TYPE - ENDTYPE defines a derived data type.
DAD¢ is declared to be a pointer to a structure -DAD- of type person.
RALPH is a normal static person, there are no pointers associated with RALPH.

To use DAD it is necessary to

```
ALLOCATE (DAD¢)
```

This will create a PERSON "on the heap" and assign a "pointer value" to DAD¢

DAD may be used in the normal way

```
DAD % AGE = 39  or  DAD % NEXTGUY¢ =  .NULL.
```

(where .NULL. is a special "unassigned" value for pointer)

```
or ALLOCATE (DAD % NEXTGUY¢)
```

The proposal will also replace allocatable arrays with pointers.

```
P¢, Q¢: POINTER (REAL  (:,:))
```

declares P¢ and Q¢ as pointers to 2 dimensional arrays and implicitly declares

P and Q as rank 2 arrays. The arrays would be created via ALLOCATE statements.

```
e.g.    ALLOCATE (P(10,10))
        ALLOCATE (Q(20,30))
```

After allocation, P and Q would be used as normal arrays

```
P  = 0
Q = 7
P = Q  (1:10,  11:20)
```

Storage would be "returned to the heap" with a FREE statement.

```
FREE (P¢)
FREE (DAD¢)
```

The pointer could be explicitly used.

```
P¢  = Q¢  or
IF (DAD¢  .EQ.  .NULL.)  STOP
```

to manipulate storage or process lists.

Comments:

1 . The pointer concept is unnecessary for most allocatable arrays. The current scheme does everything we need without introducing an un-used pointer variable.

    (ALLOCATE and FREE should accept either a pointer or an allocatable array as an argument.)

2. The "¢" shouldn't be used in ALLOCATE or FREE

3. In the example

    P¢ = Q¢

    should we require P and Q to be conformable or merely both of rank 2. The consensus seemed to be rank only.

4. A pointer to an external would be useful.

STRAW VOTES

1. Include pointers in the 8X standard with no storage association implied (20, 3, 10).

2. Define allocatable arrays without explicit user defined pointer (WG9) (11, 2, 17).

3. Use of allocatable arrays regardless of pointers (Implementation dependent) (25, 2, 7).

a.3. BIT DATA TYPE

Discussion:

Buckley	Why don't arrays of logical work?
Metcalf	Logical is word,/byte oriented
Ampt	Could be implemented as bits.
Metcalf	250,000 bits for 1 event requires efficiency. Bit is a bit.
Ampt	Don't insist on hardware, it's a two valued thing, need string I/O and conversion functions, not shift, etc.
Marusak	Users of "bits" don't think of it as "8 logical bits," it's an entity, it might be represented as a string of bits, but physicists don't think that way.
Ampt	Don't standardize hardware in the language.
Marusak	We need to help the physicist. Even if it isn't completely independent of hardware.
Crowley	F77 says logical is 1-unit of storage, can't use current logical for efficient bits.
Meek	Disagree with Kees, don't need 1 data type with 2 values. e.g. Pascal enumerated datatypes aren't just 0 and 1. Don't force logical on people who don't want it. Prefer bit data and require that it be implemented "efficiently" and be packed, etc.
Matheny	12 years ago F77 had bit like character. Discarded because couldn't be done efficiently. F-8X had it and was kicked out for "efficient" reasons. Generalized, arbitrary length string can't be done efficiently.
Buckley	Size of language? Bit just makes it bigger. Could we define a bit module and allow efficient local implementation.
Metcalf	Would be okay if they could move from vendor to vendor efficiently: SHIFT, SHIFTL, SHFTL, ...
Dahlstrand	Raster graphics needs efficient bit type. The S6 proposal likely to rally the bit users.

Hirchert Could map logical into bit in F77 unless it's in common or equivalence. But bit data type doesn't guarantee packed "efficiency". Might mean 1 bit per word.

Snoek 62 bit string not efficient on a 60 bit machine. Can't be standardized. Will need to know details of machine. Why not formulate as abstract data type, probably not the most efficient, but could be and users could pressure vendor into efficient implementation.

Adams CDC 205 bit addressable and very useful.

Ampt We know it is possible, but it won't be efficient. Perhaps an appendix describing an efficient low-level data type.

Otherwise have a set of environment inquiry functions which allow coding, of portable efficient bit operations.

Efficient bit not portable.

Crowley Bit implies "save storage," trade memory for speed. Let implementor use I bit, 1 byte, 1 word as he sees fit.

2 types of bit data type:

1) Arrays of bit things

2) Bit string (like character)

Metcalf Bit string is what we want.

Buckley Efficiency is both storage and execution.

Fortran is for science, bits are more basic than character. "Put bits in, toss character out." We don't have access to most fundamental unit the bit.

We have parameterized REAL, why not parameterized LOGICAL.

Schmitt What operation do we want on bit? Is it "logical" AND, OR or is it "integer"?

Metcalf S6, chapter 8: strings of bits; SHIFT, MASK, etc.

Paul	Want bits as "logical" and use them in array masking operation - a vector of logical. Want bit arrays, not strings. Also want to concentrate, etc. Must force as 1 bit in storage. Sections and sub-arrays of a bit array is hard to implement.
Marusak	If we had bit could naturally define byte and possibly REAL, etc. Fortran originally word oriented, invented character because we didn't have byte. Could define character in terms of bit.
Hirchert	Difference between bit string and bit array, same as between character array and string. Some comparisons give array of true or false, some a single true or false. Could define LOGICAL (LEN=) as a family like REAL (PREC=) and make it work like bit.
Snoek	Arrays and strings. Need variable length either way. Might be easier to get variable length strings.
Meek	If logical really could do what was wanted users would have forced vendors to implement it. But operations are different. People have asked for 12 years, basic facility for solid core of Fortran users.

STRAW-VOTES:

1. Do you want bit data type in the language'? (20, 1, 11)

2. Should they be implemented as bit strings? (17, 0, 12)

3. Is it sufficient to implement as a derived data type (as an application module)? (11, 7, 11)

a.4. CHARACTER FUNCTIONS EXTENSIONS

Discussion:

Buckley	Proposes adding "REV" as an optional 3rd argument to INDEX, VERIFY and ISCAN. Will cause a right-to-left scan and find last occurrence. New functions: CUT (STRING, SEQ , REV) Would normally count number of leading non-blanks - let you easily trim off trailing blanks. Returns index of last non-blank. With REV would return index of first non-blank.
Ampt	We really need varying length character, if not then CUT for sure.
Valiance	What do you return for a string of all blanks? Best answer is 1. Answer: Use MAX (1, CUT(...))
Shen	A function to reverse a string is needed. PL/I proposal in 1982 added a starting point find the first blank then easily find the next blank without needing to do the index arithmetic.
Hirchert	I think CUT is VERIFY(, , REV) with " " as the sequence.
Answer	Could be
Adams	X3J3 has decided to not accept any new proposals. We are out of time for F-8X. We need: 1) List of things that are urgently needed; 2) List of things that should be removed. Answer: Extremely unfortunate that X3J3 closes doors JUST when international community getting a chance to provide input.
Adams	We've got foreign members and meet with WG/9 every 1-2 years.
Ampt	Not all Europeans can easily go to X3J3 meetings
STRAW VOTE	X3J3 should insure that facility to obtain nonblank length is in the language? (28, 0, 2)

a.5 ENTITY ORIENTED DECLARATION

Hirchert	Entity Oriented Declaration are an alternative way to declare attributes.
Buckley	Take them out. They would be better if we started from scratch. We aren't. They are redundant.
Paul	Kurt's example doesn't show problems. The declarations are long and verbose. We have deprecated current declarations, implies major compatibility problem for 9X.
DuCroz	Likes them, people use them.
Pollicini	Likes these. But we should have only 1 way to do it in standard.
Vallance	Easy to do automatic conversion.
STRAW VOTES	1. Should entity oriented declaration be kept in F8X (9, 13, 11). 2. Variable Length character. Should they be included in F8X (14, 3, 15).

b. ARRAYS

References: B-4, The Proposed Vector/Array Extensions.

The presentation consisted of a review of extensions related to array data types. The presentation included a discussion of array sections, array-valued functions and new statements for operation oil arrays.

Discussion:

Question Why is there not a matrix multiply operator? (Rather than an intrinsic function?)

Answer (Paul) There are problems with the character set -- is a lack of available new, suitable symbols. It was further pointed out that there are now new facilities to define or override operators, so that users can DEFINE a matrix multiply operator if they wish.

Question (Suggestion) The document must do a better job of explaining automatic arrays, assumed size, adjustable arrays etc. There is considerable confusion in the existing terminology.

Question In the use of elemental functions -- where is a scalar expanded, on the calling side or in the function?

Answer 1. In the caller.

2. Elemental functions are intrinsic (users cannot define them), so the compilers will know what to do.

Question What about duplication of indices in vector-valued subscripts? This hinders portability, in that the result is processor-dependent.

Answer Many people (on the X3J3 Committee) dislike this as well, but feel that the functionality outweighs the disadvantages. The problems arise in STORES, not in FETCHES. In the discussion that followed, the suggestion was made not to allow vector-valued subscripts on the left side of the equals sign(=). Many-to-one mapping through IDENTIFY, for example, is not allowed on the left side of the equals sign(=)

Question (observation) If one wants to shrink the size of the language, a good candidate is the block form of the WHERE construct, especially the OTHERWISE, Or perhaps remove the entire construct.

STRAW VOTES

1. Shall the Block WHERE construct as it currently exists in Fortran 8X be retained? (11-7-12)

2. Shall vector-valued subscripts be allowed on the left of an equals sign (=)? (11-5-15)

3. Shall FORTRAN 8X include an algorithm that defines the store sequence into a multi-valued vector-subscripted array? (15-5-12)

4. Shall the many-to-one vector subscript be prohibited on the left of the equals sign (=)? (13-5-13).

III. STATEMENT CONCEPTS

References: B-5, ANSI 8X Statements
Discussion:

Comment: (The Canadian Community) Wishes to present to X3J3 a document on relatively minor concerns: for example, the use of END DO versus REPEAT in the Block DO. Wants a DO-WHILE construct in the language, even though functionally it is already there. Feels that X3J3 should consider the use of the colon symbol in the Block DO construct.

Question	Has X3J3 considered the use of variables in FORMAT statements -- for example, N(15), where N is a variable?
Answer (Matheny)	One can achieve this with CHARACTER strings.
Comment	There are very mixed feelings about the new Source Form; the Metcalf (SEAS) survey found 13 for and 13 against. Particularly intense was the feeling about multiple statements per line.
STRAW VOTE	Shall Fortran 8X allow multiple statements per line? (9-12-5).
Question	Has any thought been given to handling color (on terminals)?
Answer	It is not obvious that the Fortran language should concern itself with this. Once again, there is the problem with a limited character set (even the full ASCII set).
Question	Concerning Character Sets -- Will keyboards be able to interchange upper/lower case?
Answer	Yes
STRAW VOTE	Do we approve of the new source form as to allowing statements to begin before column 7? (20-3-7)
Comment	Source should be rigidly structured -- what goes into the terminal should go through a syntax-directed editor that is, input to a TERMINAL may be free-form, but input to the COMPILER should be rigidly structured. The biggest cost of software is in maintenance, and that requires reading code, and that requires structure.
STRAW VOTE	1. Should Fortran 8X include a DO-WHILE construct? (7-16-9)
	2. Shall Fortran 8X include significant blanks? (16-8-7)

IV. EVENT HANDLING

References: C-20, Event Handling in Fortran
Presenter: Koblitz

Discussion:

Dahlstrand How are overflows handled?

Ans (Koblitz) Overflows are covered by the current proposals but it is not possible to continue in a way that might be desired. A handler may pass control to the next Fortran statement or may RETURNUP (or may STOP).

Dahlstrand This is clumsier than the old-fashioned CALL OVCHK. Control has to leave the environment so that values of variables cannot be changed.

Hirchert The EWICS/TC1 work is too general. An alternative proposal is being put forward by X3J3 subgroup 7/'8. This addresses itself to a smaller problem. Everything related to a possible event can be determined statically at compile time. The granularity of determination of the event can be defined. This proposal is amongst the papers in the pre-meeting, distribution for the May X3J3 meeting (paper 90(7) KWH-1).

Paul The proposal cannot deal with matrix arithmetic, only scalar arithmetic.

Hirchert The intent is not to resume an operation which has failed. It is more an escape mechanism.

Marusak I sense frustration in EWICS at X3J3's reaction to proposals. Resumption after an event is the stumbling block. I suggest a straw vote on whether it makes sense to proceed after an event.

Ampt Hirchert's proposal will not be upwards compatible. The EWICS proposal is more general. A programmer should know that a problem exists whether a fix-up is available or not. We have to -recognize that there will be no handler in many real-world situations. The requirement that resumption from a handler is at the next Fortran statement is a mistake by X3J3; resumption has to be at the leave-off point. ACTIVATE /DEACTIVATE and SUSPEND cater for all possibilities, provided there is a decent RESUME.

Answer (To Marusak) Yes, we are frustrated because we want to do real time and parallel processing and we are cut down to dealing with overflow.

Marusak	X3J3 asked you to write an overflow handler and rejected that too.
Snoek	EWICS TC1 is not writing a general overflow handler. We are not writing for the 3-second student program. We are writing for foreground or real-time mode where a program has been running for an hour and can go on to get results after an hour and twenty minutes; this is preferable to rerunning for an hour. Event handling costs something; you pay something but you get back more.
Dahlstrand	I would draw attention to “Program Structures for Exceptional Condition Handling" by Roy Levin, Carnegie -Mellon University, June 1977, NTIS report AD/A043449. There is a summary in my paper for this meeting (Paper 5). Event handling seems to be too much for Fortran 8X. I suggest it is left for 9X.
Hirchert	Event-handling increases costs by a factor of 2 or 3. It is too expensive. I see no long-term solution.
Ampt	That is the basic problem with X3J3.
Hirchert	The EWICS proposals are unacceptable from a performance point of view. Multi-tasking and exception handling should be separated. The EWICS model is inappropriate for exception handling.
Answer	Consider the simple loop: read-check input-read-check input etc. The reading and checking could be done in parallel instead of waiting for the read. I propose a straw vote on interest in the possibility of parallel processing in Fortran.
Martin	Whether people want it or not, parallel processing will be provided by suppliers. There was a workshop on programming the next generation of supercomputers at Albuquerque on February 27 and 28. It was clear that users and suppliers of large computers do not want language features yet. They want to be able to associate names with threads of control, but do not want synchronization, forking, joining or other primitives.
Ans	DIS 7846 (IRTF) does not mention hardware. it uses an abstract system and is not implementation dependent. It is possible for control to run sequentially or for there to be parallel tasks.
Hendrickson	This is like motherhood - it is hard to vote against but it is premature to standardize now. We need to get experience, then standardize,

just like the Fortran language itself in the 1950's and 60's. It may inhibit development to standardize now.

Answer We are talking at different levels. We want the facility.

Hendrickson But what about the details? Can multiprocessors access the same variables?

Answer DIS 7846 is written at the task level. It deals with communication of programs.

Hendrickson That is not what people mean by multitasking.

Ampt The programmer should have the facility in the language to organize execution of pieces of code in parallel.

Hendrickson What size piece?

Answer That is a matter for another straw vote.

Hirchert There is a difference between allowed parallelism and required parallelism, that is what can be parallel and what must be parallel, as in IRTF. We have now implicit parallelism, e.g. array processing, in Fortran 8X but there are still many unresolved problems.

Schmitt The basic problem is: at what point of time can an event occur? If it is only between statements, there is no reason for special language features. IRTF shows it can be done. Otherwise, if events can occur during execution of statements then language features are needed.

Paul The Albuquerque meeting showed that people working in the field for up to 15 years do not know what features are really needed. Ada fails in this. People know what they want but do not know how, to do it.

DuCroz If no one can point to a model that works, the area should be dropped.

Ampt You are saying that if we do not know the end of the road, we should not start the journey.

Marusak Tightly coupled processors are different from loosely coupled processors. In Albuquerque the discussion was confined to tightly coupled processors. Forking and joining can be done on loosely coupled systems by subprogram calls so there is no need for new syntax. How does one deal with synchronous handling of data, shared data? Do not add to the language now for fear of inhibiting progress.

Crowley

Consider the granularity. One line is not one task, the compiler could cope with that. If 20 to 50 lines were a task we could use a subroutine call. We can experiment now by having run time libraries, and standardize after gaining experience.

Schmitt

There exist languages now with event handling, for example for micros reading tapes, Ada, PL/I, even some versions of Basic, so it is not right to say there is no experience.

Hendrickson

Two points. First, not every user is free to use library routines. On the Cray we have to revamp the procedure calling sequence in order to use a stack. Second, there is definitely a need to split memory so that different tasks can access memory.

Matheny

This discussion makes the subject sound something new. It is not new. It has been possible to time slice for many years, consider for example the Univac 1108 Exec 8 fork and join. The only problems are to do with record-lock, consideration of which was rejected by X3J3. This is not a language problem.

Martin

The organizers of the Albuquerque meeting invited representatives of IRTF. It was unfortunate that they were unable to be present.

Hirchert

We have been discussing not event handling but multitasking for the last half hour. There is obviously disagreement over the correct model. Even for loosely coupled processors there are no widely accepted models.

Snoek

EWICS has discussed "global common" for the last two years but it is not in the papers for X3J3 and is not at the point of being put forward.

Marusak

The Albuquerque discussion was restrictive: it considered only processors in close physical proximity. X3J3 and EWICS can deal only with loosely coupled processors.

Hirchert

Even the loosely coupled processor problem is ill-defined.

Snoek

I would draw attention to my paper (paper 13) in the documents for this meeting which discusses possible functionalities for the model in chapter 19 of S7 which has as yet no useful functionality. The requirement of X3J3, to return from a handler to the next Fortran Statement affects the entire model. EWICS/TC1 has erred in not making clear to X3J3 the possible functionality of its model.

The session was adjourned and continued on the afternoon of Thursday April 12.

Ans I suggest that after this delay and at this time of the day it makes no sense to take the straw votes we suggested on Tuesday.

Snoek I agree. I very much regret that event handling has not been removed from the current deadlock by WG9. I suggest a general straw vote.

STRAW VOTE Should work continue to allow Fortran 8X users to have access to event handling? (20-0-3)

V. UNIT CONCEPTS

References: B-6, Unit Concepts.

Presenter: Crowley

Discussion:

Question	Is "private" limited?
Answer	Yes.
Question	How are bit-constants to be declared?
Answer	Probably by means of functions or by defining an extended assignment (Bitvar = Charconst), etc.
Question	What is allowed as operators?
Answer	All those that are there in Fortran or letters, but you cannot overload an operator for its current types. Chair objects to now allowing operators like tt, etc.
Hendrickson/Wagener	You can define different types of arguments for a coercion, eg. integer to bit, character to bit, etc.
Muxworthy	Doubts if an abstract data-type can be efficient.
Answer	It may be not as bad as you think. The subroutines may be in assembler.
Wagener	Most routines indeed will have faster solutions available. However, should the module be standardized, it is allowed to be implemented as efficient as a machine can do it. The advantage is that the programs using it are portable, also to machines that did not implement it that way. Besides, there is nothing that prevents in-line expansions of functions from a module.
Muxworthy	How can you standardize a module?
Wagener	X3J3 can do it. Bit datatype is high on my list for that. Furthermore, anybody can propose a standard through ANSI or directly through ISO. It would result in a collateral standard.
Buckley	The discussion on modules is relevant. Don't discuss bit again now.

Hendrickson We think of modules as definitions of intrinsic functions. It does not necessarily have to be described as an implementation. But how?

Answer The public routines of the module should return always the same. The rest can be implemented as you wish.

Wagener Any proposal for an implementational description has to be accompanied by a functional description. It is the functional description that is the more important. The implementational description just gives one way of doing it.

Hendrickson For bit that is OK, because it is sufficiently discrete. But for e.g. a matrix inversion the answer would be radically different if you allow implementors to do what they want.

Hirchert The functional description is what should be standardized. We cannot force an implementor to do his job well. The users should try to do so. As a means of documenting e.g. IRTF or GKS could be such modules. So probably other standardizing committees may be the most likely ones to write those collateral standards.

Crowley The example is just to show how an individual user could use the facilities we put in the language to write his own derived datatype

Metcalf The example (bit datatype) is rather unhappy, for it rekindles the discussion on bit-data type.

Marusak There are much better examples indeed. e.g. Modules are to replace common. They are highly useful to replace a lot of deprecated features.

Muxworthy The example is good, because it shows some of the problems, e.g. efficiency. This can only be solved by standardized modules, proposed by X3J3 soon, preventing other groups to come up with all kinds of different solutions. Furthermore, I like more general operators. It is essential to be able to define operator symbols of at least two symbols like XX X/, etc. You used priority rules for operators.

Answer X3J3 discussed priority rules, but could not resolve it, for it seemed to be too complicated. Bit datatype was voted on yesterday. Should we have a straw-vote about the multi-symbol operator?

Muxworthy No. Nobody here has experience with it, so nobody has a biased opinion.

Wagener	See the FIB. Refer to Alex. By far the major use of modules will be to replace common. The second purpose will be to define global data-structure definitions. Thirdly, e.g. defining new operators on new data-structures (like Codasyl databases.) Fourth, procedure interfaces to allow the compiler to check at compile time.
Crowley	What about intrinsic functions?
Tony	The rewrite of CH 13 and 14 is a big improvement. Go on this way. Ch. 14 should be an alphabetical list with short descriptions.
Crowley	X3J3 was still not happy with it as a finish product and gave me detailed comments.
Marusak	We started putting in the functionality. Now we are trying to make it readable. Any suggestions are welcome.
Hirchert	X3J3 took a straw-vote on how to order this material.
Muxworthy	About matrix transpose. I only want those functions that can be implemented efficiently. Can it?
Hirchert	In a smart processor: yes. X3J3 SG6 is going to propose it differently anyway.
Wilson	The list of intrinsics is growing constantly. Is there a mechanism to decide how many there should be and which ones? There is no RANDOM and no ERROR.
STRAW VOTE	Do we prefer the rewrite in Paper 9 to Chapter 14 in S7? (14,0,11)
Crowley	We are trying to reduce the size of each functional section. Probably the array stuff has most.
Hirchert	RANDOM should be rather an intrinsic subroutine. We have a precedent for that now, so it can come in now.
Tony	How about-dependent compilation?
Hirchert	We have only the restricted form of the modules.
Tony	How about inherited precision?
Hirchert	Depends on your implementation, but it can be done cleverly.

Metcalf	Warning: Ada is useless to build subroutine libraries because of such problems
Ulrich	You need only one precision as soon as the product is available.
Marusak	The current proposal deals with both evaluations, algorithms and using them.
Hirchert	Do not agree. Follows a discussion between Alex and Kurt

VI. GENERAL CONCEPTS

References: B-7, General Concepts. The issues related to the definition of terms, scope and classes of symbolic names and deprecated features were discussed. Presenter: Wagener

a. DEFINITIONS

Discussion:

Hirchert	I don't like "user-defined" since we don't recognize a "user," say "program."
	We use "defined" to describe what processor does to assign a value. Extending it would be confusing.
Shen	Should we have a specific value for "undefined" (like .NULL. for pointer)?
Answer	PRESENT is similar to this. Would like a value that could be set and checked for.
Crowley	For integer, for example, there are no free bit patterns. Term "defined" is poor word to use for "doesn't have a valid value." Confusing English.
	"Defined" also means "specified" or "declared" in English.
Snoek	The 3 examples are all valid use of "defined". A variable is "defined" if it has a value. A procedure is "defined" if it has a valid source.
Wagener	In part difference is compile-time vs. run-time. Defining a value seems different from defining a procedure.
Snoek	Doesn't feel that it is impossible to make "undefined" detectable. Merely reserve a bit pattern. Answer: Won't work for existing programs.
Hirchert	In F77, the standard "defines" things; the program "specifies" things. This is reasonably consistent in F77.
	Some processors "sort-of" can assign an undefined value. BUT it is often extremely difficult to detect. We should not require a processor to detect it.
Crowley	In F77 an "undefined" variable may not be used. However, a "processor dependent" value is not "defined" by the standard but may be used. Need different terms.

b. ARRAY ARGUMENT ASSOCIATION

Du Croz	"An array becomes defined when all of its elements become defined." Can I use only "upper triangle?" Answer: In theory a problem if INTENT IN.
Du Croz	S7 seems to be best attempt at compatibility.
Buckley	#1 requires recompilation of all program libraries.
Snoek	Yes, we'll have to recompile but get benefits - especially no storage association.
Hirchert	#1 not upwards acceptable; #3 not functional; #2 doesn't describe S7. S7 says "use dope vectors unless actual argument is continuous and dummy is not assumed shape.". Also: Recompilation is hard to manage in University environment. New rules require recompilation since old routines don't tell loader" which they are. Could we allow "storage association" call and new call.
Crowley	S7 doesn't specify implementation but says old combination must work e.g. array element -- array. For sections, etc., as actual arguments we need new rules. In effect can't use "storage association" if pass a section to an array, implies a dope vector needed. Also must match rank if passing into an assumed shape array. Expect to eliminate storage association in F9X.
Snoek	S7 as described by Ted seems to be best choice now.
Hirchert	Now, if a dummy argument is a constant array, it could have a section as an actual argument. Since it is possible there must be run-time overhead. All dummy argument arrays require dope vectors and hence recompilation.
Surdi	Why not require that an argument which might get a section be declared as assumed-shape?
Answer	Implies all dummy arrays will be declared assumed shape.
Surdi	#2 guarantees upward compatibility from F77 source and probably object code as well.

Wagener	Re-word #1 and then it seems to match S7. Requires "expanded association mechanism" involving "in part passing dope vector."
STRAW VOTE	Do you favor:
	Proposal #1 - 11
	Proposal #2 - 5
	Proposal #3 - 0
	Undecided - 4

c. VARIABLES

Discussion:

Snoek	Obviously, a variable is anything which can change it's value.
Marusak	We would qualify "array variable" and "scalar variable."
Crowley	Variables, to me, are atomic. Therefore an array can't be atomic if an element is.
Hirchert	"Variable" in F77 is fairly useless. We need to be able to describe a thing which can be "set", i.e. Jerry's definition. Also need to be able to describe objects which have a simple name - e.g. "variables and arrays."
Clarke	F77 uses variable as a restricted case. Prefers current S7 version.
Hirchert	In S7 "variable plus..." is hard to read.
Crowley	Prefer "assignable object" or "definable object."
STRAW VOTES	1. A variable is any object that can appear on the left side of an assignment statement and may be either scalar valued or array valued (12, 6, 5) 2. The term "variable" needs to be changed to include arrays. (15, 1, 4)

d. DATA STRUCTURES

Discussion:

Buckley	Common is deprecated, don't extend deprecated features.
Hirchert	Common is there, we must consider interaction.
Wagener	Also can't allow function entry to be structure valued.
STRAW VOTE	Don't allow any derived data type objects in COMMON (16, 0, 6)

e. DEPRECATED FEATURES

Discussion:

Snoek	Could also add FORMAT to deprecated list.
Answer	Difficult to know where to draw the line
Buckley	Difficult to put strings in character formats.
Answer	Could use " and ' as delimiters
Mas	Useful for several uses of same FORMAT.
Hirchert	A "character string" not likely to be checked at compile time, as optimized FORMATs often are.
STRAW VOTE	Should any F77 features be deprecated? (21, 0, 3)
Shen	Need to justify each item in list, but not necessary to put into your 3 groups.
Answer	S7 tries to do that, but by group.
Snoek	Not important, it's merely how to present to an audience.
STRAW VOTE	Should FORMAT be deprecated? (2, 13, 7)
Du Croz	Shouldn't DPROD be deprecated.
Hirchert	Need to generalize to general precision.
Shen	What if there are name conflicts in modules?
Answer	Error, unless you use the rename part of USE. How are aggregates equivalent? Same name or same structure.
Answer	Must be from same TYPE definition. To get "equivalent" types into 2 or more programs required MODULE and USE.
Buckley	Scopes. Don't use word "scope." It confuses when: A) NAME is known; B) Entity has a value; C) Object is accessible.
	Fortran is very different from block-structured languages and our use is somewhat different from common usage.

Answer	Yes, we don't have good handle on terms.
Snoek	EWICS TC1 spent a lot of time discussing scope. We think we understand part of it.
Snoek	Deprecating, "source form" is very controversial. Probably not acceptable.
Answer	In X3J3 controversy was over new source form.
Snoek	Doesn't want to have to specify which he uses.
Metcalf	If we have new, we must deprecate old.
Matheny	We tried to make old a subset of new and couldn't. Column 72 is the hard part.
Marusak	There are still reservations about new source form. Personally, I think this will prevent F8X from being accepted.
Wagener	Source form is only incompatibility between F77 and F8X. X3J3 did a similar thing, for Hollerith. We think we must do this. if public comment is negative we might have to change.
Hirchert	As many will be unhappy if we don't do something
Matheny	F-66 Hollerith was very weak. What most people used was a vendor extended Hollerith.
STRAW VOTE	Should F77 source form be deprecated. (14, 3, 5)

REFERENCES: B-8, GKS Fortran Binding

Discussion:

Snoek On behalf of NNI, I would like to make two comments:

1) This binding demonstrates the error of not providing varying length character strings. We should not continue this error in 8X

Wagener Much of the problem seems to result from using the subset, which has no length available, rather than from the lack of a varying length.

Snoek 2) The DATA statements defining the values of the enumerated type constants should not be required and these should be in an appendix, not the main body of the standard.

Hirchert I have several points:

1. Is CHARACTER *(*) allowed in the subset? The binding appears to be using it there.

2. The functional compression you suggested could have been done with additional enumerated types rather than character strings in order to avoid performance problems. I believe the reason that this was not done is that these functions are not so compressed in GKS and they wished to preserve a 1-1 correspondence between GKS functions and FORTRAN procedures.

3. If only part of your CHARACTER variable is relevant, you can keep the lengths right by passing the appropriate substring of the variable. This may not be as convenient as a varying length character string, but it certainly isn't impossible.

4. The DATA statements are not required in this version of the binding.

Metcalf We need to take some straw votes whose results I can pass on to J. Martin at the SC5 conveners meeting. How about one on whether we regret the emphasis on the subset in this binding?

Buckley I also was going to point out that the DATA statements are not required. Should we suggest that separate bindings to subset and full FORTRAN 77?

Muxworthy It is my understanding that the vote was close between DATA and PARAMETER for the enumerated constants, so we might be able to sway WG2.

DuCroz There are GKS implementations which are complete now, except for completing the external interface, so binding to the subset could be an issue. What is the status of GKS? I would hate to see it significantly delayed by our comments.

Muxworthy GKS is a separate issue from this binding. I believe that it would already be an ISO standard, were it not for clerical errors in its processing. This binding is a relatively recent work item, but WG 2 hopes to process it quickly.

Wagener I checked the standard CHARACTER*(*) is not allowed in the subset.

Snoek Having to keep track of the effective length of a CHARACTER variable in order to pass the right substring is much too inconvenient. I would like a straw vote on moving the DATA statement to an appendix.

Muxworthy With the error in the subset binding about CHARACTER*(*) WG2 will have to do some rewriting anyway.

STRAW VOTE 1. We regret the emphasis in the GKS binding on Subset Fortran 77 at the expense of full language features (19-2-2).

2. The DATA statement in the GKS binding should be replaced by parameter statements and moved to an appendix (18-0-4).

Muxworthy We also should point out the CHARACTER (*) error in the subset routines.

Du Croz The document seems to acknowledge this on Page 5.

Metcalf Yes, but look at page 50 for an example of CHARACTER *(*) being used in a routine intended for the subset.

Shen We are talking about the FORTRAN binding of GKS, but do not have GKS itself to reference.

Muxworthy The document is huge. The BSI version costs something like 26 pounds.

14. COMMENTS FROM THE DELEGATES

a. Fortran Positions of the Canadian Working Group

References: C-4, Canadian Working Group on Fortran Positions

b. Comments raised by UK Fortran Community

References: B 9, some comments raised by UK Fortran Community

c. Paged I/O

References: B-10, Paged I/O

Presenter: Clarke

Discussion:

Paul	What if X is an Array and FORMAT was longer than the space left?
Clarke	Begin printing array at that point. Wrapping is undefined
Matheny	Intrinsic function in the I/O list itself is preferable to the FORMAT statement.
Clarke	Yes
Buckley	READ row 14 Col 21? Does it go there and reads what on the screen at that point
Clarke	Yes. Intention is different than what we intended on the carriage control. That is still used.
Snoek	There should be a lot more in this direct real full screen I/O. The position cannot be read or written. Setting left/right/top/bottom margins. All these things are highly desirable. I agree that it will not be portable in the next revision. This will let you do most of the items.
Buckley	There is a full screen I/O module CDC operating system can support three terminals.
Matheny	CODASYL has come up with a standard. We are dealing with a colateral standard.

15. PRESENTATION BY THE DELEGATES

a. RESULTS FROM SEAS QUESTIONNAIRE

References: C-1 Fortran Survey
Presenter: Metcalf

Discussion:

Snoek	Do you have an indication what kind of users were involved in the survey?
Metcalf	IBM. It may not be representative
Dahlstrand	Is KEYWORD crucial and require dependent completion
Hirchert	INTERFACE BLOCK does not require dependent compilation
DuCroz	We really need enhanced CALL facility (optional arguments with checking). We sometimes have 20 arguments where 10 are optional.
Matheny	Push came from data base people
Buckley	Intrinsic functions can have optional arguments but user defined cannot.
Clarke	What is meant by conversational I/O?
Metcalf	Interaction with terminal
Ampt	User friendly is to let the computer do the checking. We have not gone far enough.
Buckley	Having optional arguments in intrinsic functions is independent of optional arguments in user defined functions
STRAW VOTE	1. Should there be optional arguments in enhanced CALL? (28, 0, 3).
	2. Should there be keyword argument? (24, 1, 6)
Buckley	Happy with recursion being in Fortran. Recursive functions must be explicitly stated.

Dahlstrand It should be a candidate for extension modules. Some need it, others don't.

Paul Have found uses but not in Fortran. Appears in several implementations today. Has not been thoroughly investigated in conjunction with other extensions. I don't fully understand or looked at synchronization features.

Hendrickson There are severe problems associated with entry points and passing of function names.

Hirchert There are three costs: 1) Local variables obtained dynamically; 2) Loss of optimization; 3) Intelligibility.

STRAW VOTE 1. Do we want recursion in the language? (12, 5, 13)

2. Favor of a language extension module (8, 12, 9)

Hendrickson Extension module would create Fortran subsets. Now I would vote no.

Buckley I support Dick's position. Application modules can be written and used by anyone.

STRAW VOTE

Recursion in the core	13
Language extension module	4
No recursion	4
Undecided	10

b. COMMENTS ON X3J3 FROM SWEDEN

References: C-6 Comments on X3J3 from Sweden

c. ECMA QUESTIONNAIRE.

References: C-3, ECMA Questionnaire.

B-11, ECMA questionnaire.

Discussion:

Dahlstrand	Company replies are very similar. Data structure was different. There was a cool response here.
Shen	There was a similar survey in the United States.
Adams	Comments will be obtained from the FORTEC/SIGNUM FIB.
Paul	We should get more questions out to the user community.
Adams	Questionnaire is really good. We get both positive and negative responses.
Ampt	Ideal questionnaire does not exist. If its too long, ignored. If it's too short it's-hard to understand. Send out questionnaire as long as you ignore the results.
Marusak	Have someone who knows how to take surveys do it. Let's have a good one.
Adams	It’s not possible to ignore comments on the FIB since each one must be answered.
Hirchert	Survey assumes user understands the feature without specifying the productions.
Paul	There's a problem with FIB syntactic definitions. Doing a survey would not be meaningful.
Shen	Publishing 8X would automatically get your responses

d. FORTRAN PROCESSOR REQUIREMENTS.

References: B-12, Fortran Processor Requirements
C-11, Conformance to the standard.

Presenter: Meek
Discussion:

Matheny	What if these requirements were to be included in the standard.
Meek	Would be delighted. Having a second standard (if not included in ANSI standard) is not enough.
Shen	ANSI standard should be thorough enough to not require a second standard. A second standard does not serve either the implementor or the user. Problems should be resolved using the standard. 77 Standard is too verbose. Final documentation should be useful and unambiguous in every respect.
Hirchert	Reid's proposal is overly strong. Requirement 3 would be exceedingly hard to state. It would be desirable as an option.
Ampt	Like the ideas expressed in Reid's proposal. There are no good solutions to questions such as: 1) What action is taken when error is detected?; 2) Where is the error detected?
Muxworthy	I'm sympathetic with this paper. However, I feel it is doomed because of environmental implications and cost to implement. A second standard is the right approach.
Paul	Requirement 2 is much too vague.
Meek	Requirement doesn't say where error is to be detected. No assumptions are made as to how a processor is to implement the requirement.
Ampt	British standard is good idea and we should continue along this line. We should make it an international standard without any references to ANSI.
Meek	There are no restrictions in its use. There is the danger of the wasting time in getting it accepted as an international standard. We should see if there is any interest in it becoming an international standard. BSI only tells how to define vendor specifications.
Shen	Can the BS metalanguage be used in 8X?

Meek Would a single metalanguage be defined for all languages. BS is not quite up to this task.

Dahlstrand The question is safety versus speed. The cost to check everything is high. There should be an option to check or not to check.

Schmitt Giving a message is not very useful

Adams There's a difference between BS and a conformity module. In the future 8X needs a conformity module.

Meek The difficulty is how to make the language modular. Conformity checks must be built in the design of the processor rather than as library routines. The only way 8X can include conformity is to go through and specify what the processor must do at each point. Part 1 of the standard takes on the conventional form. Part 2 defines the processor requirements and Part 3 are additional requirement. Section 6 in the Fortran Processor Requirements (Reference B-12). Each part should be separated out or else the standard would be unreadable. We could produce a Fortran 77 processor specification standard from the BSI standard.

Du Croz I support removing any ambiguities from the standard. There should be recommendations as to what the processor should do for the ambiguities. We need to tighten up the loose ends. 8X should have an appendix to define what the processor should do.

Matheny Extensions are required for new hardware. Standardization of extension misses the point.

Meek It is desirable to allow core plus module to allow a consistent way to meet this requirement. There are two answers to Jeremy's question: 1) The only way to get agreement on the standard is to purposely leave certain items vague; 2) If standard is permissive the user can voluntarily be restrictive in its use. User builds specification to his requirements. Voluntary, not mandatory.

16. REMAINING COMMENTS AND PRESENTATIONS

a. FIB TO BE FORWARDED TO SC5 FOR COMMENTS

Discussion:

Buckley I believe IBM's comments should be forwarded with the FIB.

Adams No. I feel IBM/DEC's responses should not be publicized.

Marshall Wouldn't that cause problems with X3

Adams Yes

Hendrickson Why would WG9 want to send the FIB out if it's a X3J3 document?

Adams We cannot send it to SC5. There is no reason.

Buckley What's the purpose of sending it to SC5?

Martin We would like it moved along at the same pace on both the national and international scene.

Adams It's usual for WG9 to send documents out for informational purposes.

Schmitt X3J3 can send it to WG9, then WG9 can put cover letter on it with a WG9 number and send it to SC5

Koblitz That will allow a wide circulation for comments

Snoek FIB should be documented without comments

Meek The FIB that is issued by X3 with an ANSI # and an official WG9 # should be distributed.

Ampt Comments are not part of the document. FIB should be distributed without comments.

Meek FIB should be whatever X3 sends with or without comments.

Adams This was discussed at X3J3 and we concluded not to include comments.

Buckley Although there are just two comments, they are from two non trivial members of X3J3.

Dahlstrand I would like to see the final document with cover letter and letter ballot.

Meek Reporting to SC5 is the responsibility of the convener. If letter ballot is required, there will be a 3 to 4 month delay. There is no need for a letter ballot. It is not a standard. The convener can decide.

Adams The instruction from ISO WG9 is to reach a consensus. Formal voting is discouraged.

Koblitz It is worth while sending the FIB without comments. Response should not be influenced by two companies.

Buckley Send it to SC5 with a letter from the convener stating that it's for information only

Schmitt Send the FIB as is

Paul The FIB is now X3 document. The only option is to withdraw it.

Ampt I want to be sure that no comments are included

STRAW VOTE Forward FIB to SC5 without comments (15, 7, 6)

Meek If X3 requires any revision the same document must be distributed in the United States and internationally. Once the FIB is approved for distribution, the convener should be authorized. Send it to SC5 with a cover letter including any comments SC5 ought to know.

Schmitt I agree with Brian. If X3 is rewriting the FIB they can include any comments but not in the same document.

Ampt It's alright for X3J3 to change the text of the FIB for clarification. But comments should not be part of the SC5 document.

Shen I don't know what value it is to have comments at this stage. Why not accompany the FIB with the complete S7 document. Several members of this WG did not receive the S7 document until after the meeting.

Martin S7 is only a working document. It is not a completed document. We will be happy to distribute it.

Wagener Purpose of the FIB is the desire to get the information out in a timely, complete and honest manner. We wanted to provide a

	document that clearly summarized the status of the language. We wanted the distribution to be as wide and as soon as possible and to get as many comments as we could. Some of the language in the comments are inflammatory and prejudiced.
Koblitz	FIB should be distributed ASAP with as few working changes as possible.
Adams	X3 official copy of the FIB should be forwarded to SC5
Meek	If the WG9 copy of the FIB should be stamped DRAFT copy and can be distributed to whomever we like.
STRAW VOTE	WG9 strongly objects to the inclusion of comments in the FIB (17, 6, 3)
Meek	Send ASAP to WG9 the FIB that has been approved in the United States. If there is a delay, request members of WG9 to distribute draft copies of the FIB.
STRAW VOTE	SC5, WG9 welcomes an official FIB on the status of work of X3J3 and charges its convener upon receipt to forward such FIB to SC5 for comments. (22, 0, 3)

b. EVENT HANDLING

STRAW VOTE: Does it make sense to continue working to find a way for Fortran 8X users to use event handling in a standardized way? (22, 0, 3)

c. DRAFT INTERNATIONAL STANDARD

Reference: C-22, Draft International Standard ISO/DIS 2382/15

Ampt	I draw your attention to Data Processing Vocabulary Part 15: Programming Language. The Fortran part should be reviewed by X3J3 and WG9. Voting terminates 8/9/84.

17. Recommendations

    None

18. REQUIREMENTS CONCERNING A SUBSEQUENT MEETING

Discussion:

Ampt About two years ago I asked that a list of inexpensive hotels be provided. Send Hotel address in Bonn including a price list. Also include recommendations

Meek The middle of June is impossible in the United Kingdom since it falls in the middle of examinations in the universities.

Adams September is the same problem as June in the United States.

Rotthaeuser 1st or 2nd week in July is better than August

STRAW VOTE

1st Week of July	- 6
2nd week of July	- 1
Either	- 15
None	- 1

Buckley In 1986 we can host a meeting at the university of Halifax. Alternatives would be Ottawa University. Residence moderately priced.

Metcalf It’s difficult to get money to travel outside Europe

Meek It would be appreciated if we held a colateral meeting with the full X3J3.

Paul We may have an ANSI meeting in Oxford. Possibly a consecutive meeting 7/1 and 7/8 in Bonn and Oxford.

19. The delegates expressed appreciation to CERN, Mike Metcalf and all others who assisted with local arrangements.

20. The meeting adjourned at 5:30 PM on Thursday, April 12, 1984.