345

March 1980 Fortran Newsletter Volume 6, Number 1, Page 1

=FOR-WORD=>

X3J3 To Consider Real-Time and Graphics Applications As Second and Third Examples of Application Modules

X3J3, the Technical Committee for Fortran under ANSI (American National Standards Institute) has maintained continuing liaison with U.S. and international Committees interested in Real-Time Applications of Fortran. At the most recent meeting of X3J3 (January 1980), an ad hoc Task Group was formed as a precursor to a formal Task Group (X3J3.2) on Control of Multi-Tasking Systems. This group will coordinate its work closely with that of the Instrument Society of America (ISA), which has been working on standards in the process-control area, and with the work of the European Workshop on Industrial Control Systems (EWICS).

Liaison with the ANSI Technical Committee on Graphics (X3H3) is also being established. In a letter to X3J3 members dated February 15, the X3J3 Chairman, Jeanne Adams, stated that formal liaison has been requested by X3H3.

If you are particularly interested in X3J3 activities in the Multi-tasking (and Process Control or Real-Time) or Graphics application areas, your inquiry addressed to Loren P. Meissner (address below) will be appropriately forwarded.

Application Modules? One scenario that appears possible for the future Fortran "Core plus Modules" architecture, is that X3J3 Task Groups would be formed to develop Application Modules that seem to require syntactic extensions. The existing Task Group X3J3.1 on Data Base applications, the proposed Task Group X3J3.2 on Multi-Tasking applications, and possibly a future Task Group on Graphics applications, might all develop proposals for Application Modules to be combined with a Core language and other Basic Modules developed by X3J3.

The Chairman's February 15 letter expressed concern as to the ways in which such application modules might be developed, controlled, and standardized. One particular area of concern is the timing of revisions to Application Modules: Must such revisions occur in synchronism with revisions of the Core (or other parts of the Standard)? This question is also related to "packaging" of the Standard document(s): How are the review procedures of ANSI to be applied to a language consisting of a Core and several different kinds of Modules?

Next Fortran in 1985? At its meeting in January 1980, X3J3 considered a revised "Draft Milestone Chart". This chart is not a concrete commitment, but a working guide giving targets for Committee work on various phases of development of the next Fortran standard. The target dates listed include completion of language feature definition early in 1982, public review beginning in mid 1983, and final ANSI action in 1985.

The Working Groups within X3J3 agreed that initial draft proposals for all major features should be submitted by mid 1980. Thus the last half of 1980 and all of 1981 is allocated for evaluating the language as a whole, studying the interactions among features, and establishing the content of the Core and of the Basic Modules.

Loren P. Meissner
CSAM - 50B
Lawrence Berkeley Laboratory
Berkeley CA 94720

FIRST CLASS

PUB 345

Fortran Data Base Features Proposed. At the January meeting of X3J3, the Task Group on Data Base Manipulation (X3J3.1) presented a draft proposal for a Data Base Application Module for the next Fortran standard. The proposal would use the current CODASYL Fortran Data Base JOD (version 2, January 1980) as a basis.

The Task Group intends to continue developing this data base facility, using an approach that involves a new "keyword" syntax modelled after the input and output statements of Fortran 77. That is, special data base statements could be added to the language, provided that they follow a syntactic form consisting of a keyword, a parenthesized "control list", and a list of data items.

This Data Base proposal was given an encouraging reception by X3J3.

X3J3 Adopts Character Extensions. At the January 1980 meeting, X3J3 considered seven proposals for extensions to Character data type, and adopted six of them for the next Fortran standard. **[Remember that X3J3 actions may be later withdrawn or extensively modified prior to final standardization.]**

1. Substrings of constants. Substring notation may be applied to a character constant, including a symbolic name of a character constant.

2. Assignment overlap. Character positions mentioned on the left side, in a character assignment statement, may also appear on the right. Similar proposals have already been adopted for Arrays and for Bit strings.

3. Null strings. Character entities, including constants, may have length zero (but not negative length).

4. Null substrings. A substring specifier may have an upper bound that is one position to the left of the lower bound, thus specifying zero length.

5. Concatenation of passed-length strings. The Fortran 77 restriction, that passed-length strings must not be concatenated in certain contexts, is not needed in the next standard, since limited dynamic storage availibility is assumed.

6. Intrinsics for ASCII conversion. Intrinsics analogous to the Fortran 77 functions CHAR and ICHAR, but based on the ASCII collating sequence, were adopted. The new functions are to be called ACHAR and IACHAR.

7. Additional Intrinsics. Several additional intrinsic functions for character manipulation were proposed. However, this proposal was later withdrawn for further study.

Core and Modules Architecture. A formal proposal defining the Core plus Modules "architecture" for Fortran was adopted. Another proposal was adopted which set forth the following ten criteria for including specific features in the Core: (1) The Core must be a complete language. (2) It should be regular and consistent. (3) It should be concise. (4) It should be portable. (5) It should have a minimum of duplication of functionality. (6) It should be capable of efficient implementation. (7) It should use modern language technology. (8) It should be broadly acceptable. (9) It should be implementable on a wide variety of machines. (10) It must be compatible with the Obsolete Features Module.

Other Items Discussed by X3J3. The Committee discussed Varying-length Character data; Compile-time facilities including compile-time variables, compile-time intrinsics, and compile-time control constructs; Generalized specification of Precision; and some extensions to Format edit descriptors to permit zero values for parameters such as repeat count, Tab count, or field width.

CORRESPONDENCE

Significant Blanks. In the previous issue of For-Word (Volume 5, Number 4), readers were asked to indicate a preference concerning whether blanks should be made significant in Fortran. At press time for this issue, 50 responses had been received. 40 of these favor a change to significant blanks in the next standard. 6 are "Strongly opposed to significant blanks, ever". Two are "Strongly opposed to significant blanks in the next standard", and two are in favor of a change eventually, but not in the next standard. Nobody was "Uncertain as to whether a change should be made".

These results certainly indicate a sentiment in favor of changing to significant blanks. Perhaps only slightly less notable, however, was the polarization of opinion on the issue: None of the responses indicated the least vacillation or indecision. This prolarization was reflected in the comments reprinted below.

"I am strongly opposed to significant blanks, ever. Blanks can improve the readability of certain 'formula' statements. Misinterpretation by a program maintenace programmer or debugger can be more costly than storage gained by deleting blanks."

"Blanks, both spaces and blank lines, greatly assist presentation and readability. Good writing is a responsibility. [Also mentions an existing software tool that converts between significant and non-significant blanks.]"

"We would favour a change as soon as possible, provided that you get it right!"

"I strongly favour a change in the next standard to make blanks significant, for all of the 'good' reasons outlined in the article. However, the thought of trying to cope with the transition scares me. It will have an enormous impact. I hope (if it is to be done) that both a software tool and a liberal 'advance warning' period are provided. It would be very nice if reformatting tools could be provided as a part of the compiler."

"I feel strongly that blanks should be made significant as soon as possible (assuming that Fortran will continue to be used as a programming tool.) Some enterprising colleagues of mine who have put together some interesting and useful tools for PASCAL, and for an in-house systems language, are simply not interested in writing versions which take Fortran programs as input. After some prodding, I found out that the complexity of lexical analysis for Fortran is the deciding factor. The total amount of effort required to convert programs which don't have 'necessary' blanks would be small compared to the effort required by all future implementers of Fortran compilers and other tools. Furthermore, some tools will simply NOT be written, imposing an unreasonable burden on those hapless Fortran programmers who would be willing to type blanks if only they could use tools. Further, blanksinobviousplacesmakethingsmorereadableforhumansaswellascompilers."

"Insignificant blanks looked like a good idea 25 years ago but we have learned better. However we seem to be stuck with them. Ideally we should support both styles, but without sticking either Old-timers or New-comers with a specification line. (A specification statement would be appropriate in the Sub-Set.) The problem is not language but style. And you cannot legislate good taste. I cannot understand why anyone would prefer the insignificant style, but there must be some left. For those who use significant style, whatever is done is irrelevant."

"I am in favor of making blanks significant in the next standard. I know of no one who consciously uses the insignificant blank feature. I am always unpleasantly surprised when a compiler decides I had an insignificant blank somewhere. If this will mean that Fortran words are reserved, that is O.K. with me."

"Include significant blanks as required with Free Form Source Input. Both represent a change from current Fortran; both are desirable and easy to implement."

"The longer the process of change is delayed, the more painful it will be. S.I. Feldman has a routine for converting Fortran programs to blank-significant source."

"Blanks should be significant. Readable text is a basic element. Change should be made in the next standard."

"I am in favor of significant blanks in the next standard. However, GOTO should also be allowed as GO TO."

"The ideas about uniformity of the programming environment make sense. I like the argument."

"Concerning the question of significant blanks, I strongly favor a change in the next standard. I see no useful purpose to retaining non-significant blanks, particularly since it has been years since I saw a program that used non-significant blanks."

"I feel that Fortran should have significant blanks as soon as possible. The opinion has been developed as a result of my experience providing systems support to both Fortran and COBOL users, and my current involvement in the COBOL standardization effort."

"I'd like to see blanks made significant <u>usually</u>. I see no need to intersperse blanks in the middle of a variable name, so within a symbol I say blanks should be significant. However, I think it is important to be permitted to intersperse blanks between symbols and punctuation (like comma, plus, minus, etc.). So I argue for a rule like: A blank terminates a symbol; a blank does not terminate a statement."

"I favor a change in the next standard. I feel the change should be coordinated with, and dependent upon, the use of free form source input, which should <u>not</u> require & for continuation."

"I favor a change in the next standard to allow significant blanks. The principal use of non-significant blanks is as a separator in multi-word identifiers. I believe this function could be adequately replaced by a separator character such as the underscore."

"For reasons similar to those in the second paragraph of the article [which relates to 'packages of software tools to provide a stable, uniform environment across different languages, different machines, and different operating systems'], blank-delimited symbols have become a <u>de facto</u> standard in our environment. Further, <u>some</u> of our compilers readily accept variable-length lines and column violations. The proposal is, in my opinion, merely a recognition of reality."

"Any change to make blanks significant should include the option to have them insignificant as now, if such be the user's choice. Such a change should be made in the next standard."

"In particular, 'A = X SUB I', where XSUBI is one variable, is nice; but 'DO10I=1,13' is <u>ridiculous</u>! I <u>do</u> want to be able to surround my <u>operators</u> and equal signs with extra blanks, and put blanks after my commas."

ANNOUNCEMENTS

HPAC (Oxford, England) has implemented a full Fortran 77 compiler. A compiler switch permits selection of any of four options: Fortran 77, Fortran 66 (ANSI X3.9-1966), Fortran 77 plus Hollerith, or Fortran with Real-Time extensions. The compiler interfaces with other HPAC products, including compilers for BCPL, CORAL 66, RTL-2, BASIC, and Algol 60. HPAC products are available on Computer Automation NM4, Digital PDP-11, and Data General computers. Average cost of a compiler is under $2000. For further information, write P. J. Burns de Bono, HPAC Limited, Cherwell House, 1-5 London Place, St. Clements, Oxford OX4 1YT, England.

The University of Salford (England) has implemented a compiler which supports the Fortran 77 language in its entirety. It is available on ICL 1900 and 2900 (DME) systems. The compiler (optionally) checks for exact conformance to the Fortran 77 standard. A run time diagnostic facility, a compile time cross reference facility, and a run time profile (statement execution count) facility are provided. Write David M. Vallance, Computing Centre, University of Salford, Salford M5 4WT, Lancashire, England.

Fujitsu Ltd (Japan) announces a Fortran 77 compiler. It will run on FACOM M series computers and IBM computers with MVS operating system, and is compatible with IBM Fortran IV H Extended and G1 as for source program. Extensions include data types INTEGER*2, LOGICAL*1, REAL*16, COMPLEX*16, and COMPLEX*32; Hollerith and hexadecimal constants; some input and output extensions; DO WHILE, DO UNTIL, and extended range DO control structures; debug facilities; initialization in specification; and free-form source. For further information, telex Japan 3922508, facsimile Japan 0559-23-5330, or write to Hideo Wada, F/A section LP division, Fujitsu Ltd, 140 Miyamoto Numazu-shi Shizuoka-ken, 410-03 Japan.

Texas Instruments (Houston) has implemented ANSI X3.9-1978, in a product called DX10 Fortran-78. This implementation, for DS990 models 4 through 30, includes ISA-recommended process control extensions, a math stat library, and interfaces to packages for generalized forms, sort merge, and data base management. A one year license costs $3000.

Virtual Systems, 1500 Newell Avenue, Suite 406, Walnut Creek CA 94596, announces a Fortran 77 compiler, integrated with the VS Microbench software system which runs on PDP-11 and VAX hardware as well as Intel, Motorola, and Zilog microprocessors. A library of intrinsic functions and runtime support routines is included. Features include: ANSI compatibility; 32 bit arithmetic; arrays up to one megabite in size; in-line assembly language; user operating system interface; bitwise logical functions; and ROM-RAM support. The price is $3750 to $4250.

REFERENCES

Fortran for business people by Didday, Page, Hayen, and Wayne (West Publ. Co., NY, 1978) [See ACM Computing Reviews, July 1979] features problems and examples related to payroll, sales, balance sheets, etc. An attribute is "the subtle, conversational style of the writing". Each chapter begins by explicitly stating the chapter objectives.

CODASYL Operating System Command Language JOD. This is a preliminary document for comment and review. The aim is to develop requirements for a computer-independent operating system command and response language (OSCRL). It is expected that ANSI X3H1 will use this document as a basis for developing an OSCRL standard. Send $6.00, payable to the Receiver General for Canada, to: Materiel Data Management Branch, Dept of Supply and Services, 4th floor Core B1, Place du Portage Phase III, 11 Laurier St, Hull, Quebec, Canada K1a 0S5.

Functions for Manipulating Floating Point Numbers by John Reid (Harwell, England) appeared in the December 1979 issue of ACM SIGNUM Newsletter, pages 11 - 14. For Fortran use, this paper proposes that the names EPSLN, INTXP, and SETXP be used when possible.

Note. A correspondent points out that "there is an interesting article on recursion in Fortran, in Creative Computing, Jan. 1980, pp 98 - 102".

CONCERNING FOR-WORD

This Newsletter is prepared for the U.S. Dept. of Energy under Contract W-7405-ENG-48.

The editor's name and address appear in the Mailing area on the first page of each issue. Requests for additions or corrections to the mailing list should be directed to the editor.

Correspondence on all Fortran-related topics is welcomed. Especially solicited are reviews of recent Fortran textbooks, software products, literature, etc.

X3J3 To Consider Real-Time and Graphics Applications As Second and Third Examples of Application Modules

X3J3, the Technical Committee for Fortran under ANSI (American National Standards Institute) has maintained continuing liaison with U.S. and international Committees interested in Real-Time Applications of Fortran. At the most recent meeting of X3J3 (January 1980), an ad hoc Task Group was formed as a precursor to a formal Task Group (X3J3.2) on Control of Multi-Tasking Systems. This group will coordinate its work closely with that of the Instrument Society of America (ISA), which has been working on standards in the process-control area, and with the work of the European Workshop on Industrial Control Systems (EWICS).

Liaison with the ANSI Technical Committee on Graphics (X3H3) is also being established. In a letter to X3J3 members dated February 15, the X3J3 Chairman, Jeanne Adams, stated that formal liaison has been requested by X3H3.

If you are particularly interested in X3J3 activities in the Multi-tasking (and Process Control or Real-Time) or Graphics application areas, your inquiry addressed to Loren P. Meissner (address below) will be appropriately forwarded.

Application Modules? One scenario that appears possible for the future Fortran "Core plus Modules" architecture, is that X3J3 Task Groups would be formed to develop Application Modules that seem to require syntactic extensions. The existing Task Group X3J3.1 on Data Base applications, the proposed Task Group X3J3.2 on Multi-Tasking applications, and possibly a future Task Group on Graphics applications, might all develop proposals for Application Modules to be combined with a Core language and other Basic Modules developed by X3J3.

The Chairman's February 15 letter expressed concern as to the ways in which such application modules might be developed, controlled, and standardized. One particular area of concern is the timing of revisions to Application Modules: Must such revisions occur in synchronism with revisions of the Core (or other parts of the Standard)? This question is also related to "packaging" of the Standard document(s): How are the review procedures of ANSI to be applied to a language consisting of a Core and several different kinds of Modules?

Next Fortran in 1985? At its meeting in January 1980, X3J3 considered a revised "Draft Milestone Chart". This chart is not a concrete commitment, but a working guide giving targets for Committee work on various phases of development of the next Fortran standard. The target dates listed include completion of language feature definition early in 1982, public review beginning in mid 1983, and final ANSI action in 1985.

The Working Groups within X3J3 agreed that initial draft proposals for all major features should be submitted by mid 1980. Thus the last half of 1980 and all of 1981 is allocated for evaluating the language as a whole, studying the interactions among features, and establishing the content of the Core and of the Basic Modules.

Loren P. Meissner
CSAM - 50B
Lawrence Berkeley Laboratory
Berkeley CA 94720

F I R S T C L A S S

Fortran Data Base Features Proposed. At the January meeting of X3J3, the Task Group on Data Base Manipulation (X3J3.1) presented a draft proposal for a Data Base Application Module for the next Fortran standard. The proposal would use the current CODASYL Fortran Data Base JOD (version 2, January 1980) as a basis.

The Task Group intends to continue developing this data base facility, using an approach that involves a new "keyword" syntax modelled after the input and output statements of Fortran 77. That is, special data base statements could be added to the language, provided that they follow a syntactic form consisting of a keyword, a parenthesized "control list", and a list of data items.

This Data Base proposal was given an encouraging reception by X3J3.

X3J3 Adopts Character Extensions. At the January 1980 meeting, X3J3 considered seven proposals for extensions to Character data type, and adopted six of them for the next Fortran standard. **[Remember that X3J3 actions may be later withdrawn or extensively modified prior to final standardization.]**

1. Substrings of constants. Substring notation may be applied to a character constant, including a symbolic name of a character constant.

2. Assignment overlap. Character positions mentioned on the left side, in a character assignment statement, may also appear on the right. Similar proposals have already been adopted for Arrays and for Bit strings.

3. Null strings. Character entities, including constants, may have length zero (but not negative length).

4. Null substrings. A substring specifier may have an upper bound that is one position to the left of the lower bound, thus specifying zero length.

5. Concatenation of passed-length strings. The Fortran 77 restriction, that passed-length strings must not be concatenated in certain contexts, is not needed in the next standard, since limited dynamic storage availibility is assumed.

6. Intrinsics for ASCII conversion. Intrinsics analogous to the Fortran 77 functions CHAR and ICHAR, but based on the ASCII collating sequence, were adopted. The new functions are to be called ACHAR and IACHAR.

7. Additional Intrinsics. Several additional intrinsic functions for character manipulation were proposed. However, this proposal was later withdrawn for further study.

Core and Modules Architecture. A formal proposal defining the Core plus Modules "architecture" for Fortran was adopted. Another proposal was adopted which set forth the following ten criteria for including specific features in the Core: (1) The Core must be a complete language. (2) It should be regular and consistent. (3) It should be concise. (4) It should be portable. (5) It should have a minimum of duplication of functionality. (6) It should be capable of efficient implementation. (7) It should use modern language technology. (8) It should be broadly acceptable. (9) It should be implementable on a wide variety of machines. (10) It must be compatible with the Obsolete Features Module.

Other Items Discussed by X3J3. The Committee discussed Varying-length Character data; Compile-time facilities including compile-time variables, compile-time intrinsics, and compile-time control constructs; Generalized specification of Precision; and some extensions to Format edit descriptors to permit zero values for parameters such as repeat count, Tab count, or field width.

CORRESPONDENCE

Significant Blanks. In the previous issue of For-Word (Volume 5, Number 4), readers were asked to indicate a preference concerning whether blanks should be made significant in Fortran. At press time for this issue, 50 responses had been received. 40 of these favor a change to significant blanks in the next standard. 6 are "Strongly opposed to significant blanks, ever". Two are "Strongly opposed to significant blanks in the next standard", and two are in favor of a change eventually, but not in the next standard. Nobody was "Uncertain as to whether a change should be made".

These results certainly indicate a sentiment in favor of changing to significant blanks. Perhaps only slightly less notable, however, was the polarization of opinion on the issue: None of the responses indicated the least vacillation or indecision. This prolarization was reflected in the comments reprinted below.

"I am strongly opposed to significant blanks, ever. Blanks can improve the readability of certain 'formula' statements. Misinterpretation by a program maintenace programmer or debugger can be more costly than storage gained by deleting blanks."

"Blanks, both spaces and blank lines, greatly assist presentation and readability. Good writing is a responsibility. [Also mentions an existing software tool that converts between significant and non-significant blanks.]"

"We would favour a change as soon as possible, provided that you get it right!"

"I strongly favour a change in the next standard to make blanks significant, for all of the 'good' reasons outlined in the article. However, the thought of trying to cope with the transition scares me. It will have an enormous impact. I hope (if it is to be done) that both a software tool and a liberal 'advance warning' period are provided. It would be very nice if reformatting tools could be provided as a part of the compiler."

"I feel strongly that blanks should be made significant as soon as possible (assuming that Fortran will continue to be used as a programming tool.) Some enterprising colleagues of mine who have put together some interesting and useful tools for PASCAL, and for an in-house systems language, are simply not interested in writing versions which take Fortran programs as input. After some prodding, I found out that the complexity of lexical analysis for Fortran is the deciding factor. The total amount of effort required to convert programs which don't have 'necessary' blanks would be small compared to the effort required by all future implementers of Fortran compilers and other tools. Furthermore, some tools will simply NOT be written, imposing an unreasonable burden on those hapless Fortran programmers who would be willing to type blanks if only they could use tools. Further, blanksinobviousplacesmakethingsmorereadableforhumansaswellascompilers."

"Insignificant blanks looked like a good idea 25 years ago but we have learned better. However we seem to be stuck with them. Ideally we should support both styles, but without sticking either Old-timers or New-comers with a specification line. (A specification statement would be appropriate in the Sub-Set.) The problem is not language but style. And you cannot legislate good taste. I cannot understand why anyone would prefer the insignificant style, but there must be some left. For those who use significant style, whatever is done is irrelevant."

"I am in favor of making blanks significant in the next standard. I know of no one who consciously uses the insignificant blank feature. I am always unpleasantly surprised when a compiler decides I had an insignificant blank somewhere. If this will mean that Fortran words are reserved, that is O.K. with me."

"Include significant blanks as required with Free Form Source Input. Both represent a change from current Fortran; both are desirable and easy to implement."

"The longer the process of change is delayed, the more painful it will be. S.I. Feldman has a routine for converting Fortran programs to blank-significant source."

"Blanks should be significant. Readable text is a basic element. Change should be made in the next standard."

"I am in favor of significant blanks in the next standard. However, GOTO should also be allowed as GO TO."

"The ideas about uniformity of the programming environment make sense. I like the argument."

"Concerning the question of significant blanks, I strongly favor a change in the next standard. I see no useful purpose to retaining non-significant blanks, particularly since it has been years since I saw a program that used non-significant blanks."

"I feel that Fortran should have significant blanks as soon as possible. The opinion has been developed as a result of my experience providing systems support to both Fortran and COBOL users, and my current involvement in the COBOL standardization effort."

"I'd like to see blanks made significant <u>usually</u>. I see no need to intersperse blanks in the middle of a variable name, so within a symbol I say blanks should be significant. However, I think it is important to be permitted to intersperse blanks between symbols and punctuation (like comma, plus, minus, etc.). So I argue for a rule like: A blank terminates a symbol; a blank does not terminate a statement."

"I favor a change in the next standard. I feel the change should be coordinated with, and dependent upon, the use of free form source input, which should <u>not</u> require & for continuation."

"I favor a change in the next standard to allow significant blanks. The principal use of non-significant blanks is as a separator in multi-word identifiers. I believe this function could be adequately replaced by a separator character such as the underscore."

"For reasons similar to those in the second paragraph of the article [which relates to 'packages of software tools to provide a stable, uniform environment across different languages, different machines, and different operating systems'], blank-delimited symbols have become a <u>de facto</u> standard in our environment. Further, <u>some</u> of our compilers readily accept variable-length lines and column violations. The proposal is, in my opinion, merely a recognition of reality."

"Any change to make blanks significant should include the option to have them insignificant as now, if such be the user's choice. Such a change should be made in the next standard."

"In particular, 'A = X SUB I', where XSUBI is one variable, is nice; but 'DO10I=1,13' is <u>ridiculous</u>! I <u>do</u> want to be able to surround my <u>operators</u> and equal signs with extra blanks, and put blanks after my commas."

ANNOUNCEMENTS

HPAC (Oxford, England) has implemented a full Fortran 77 compiler. A compiler switch permits selection of any of four options: Fortran 77, Fortran 66 (ANSI X3.9-1966), Fortran 77 plus Hollerith, or Fortran with Real-Time extensions. The compiler interfaces with other HPAC products, including compilers for BCPL, CORAL 66, RTL-2, BASIC, and Algol 60. HPAC products are available on Computer Automation NM4, Digital PDP-11, and Data General computers. Average cost of a compiler is under $2000. For further information, write P. J. Burns de Bono, HPAC Limited, Cherwell House, 1-5 London Place, St. Clements, Oxford OX4 1YT, England.

The University of Salford (England) has implemented a compiler which supports the Fortran 77 language in its entirety. It is available on ICL 1900 and 2900 (DME) systems. The compiler (optionally) checks for exact conformance to the Fortran 77 standard. A run time diagnostic facility, a compile time cross reference facility, and a run time profile (statement execution count) facility are provided. Write David M. Vallance, Computing Centre, University of Salford, Salford M5 4WT, Lancashire, England.

Fujitsu Ltd (Japan) announces a Fortran 77 compiler. It will run on FACOM M series computers and IBM computers with MVS operating system, and is compatible with IBM Fortran IV H Extended and G1 as for source program. Extensions include data types INTEGER*2, LOGICAL*1, REAL*16, COMPLEX*16, and COMPLEX*32; Hollerith and hexadecimal constants; some input and output extensions; DO WHILE, DO UNTIL, and extended range DO control structures; debug facilities; initialization in specification; and free-form source. For further information, telex Japan 3922508, facsimile Japan 0559-23-5330, or write to Hideo Wada, F/A section LP division, Fujitsu Ltd, 140 Miyamoto Numazu-shi Shizuoka-ken, 410-03 Japan.

Texas Instruments (Houston) has implemented ANSI X3.9-1978, in a product called DX10 Fortran-78. This implementation, for DS990 models 4 through 30, includes ISA-recommended process control extensions, a math stat library, and interfaces to packages for generalized forms, sort merge, and data base management. A one year license costs $3000.

Virtual Systems, 1500 Newell Avenue, Suite 406, Walnut Creek CA 94596, announces a Fortran 77 compiler, integrated with the VS Microbench software system which runs on PDP-11 and VAX hardware as well as Intel, Motorola, and Zilog microprocessors. A library of intrinsic functions and runtime support routines is included. Features include: ANSI compatibility; 32 bit arithmetic; arrays up to one megabite in size; in-line assembly language; user operating system interface; bitwise logical functions; and ROM-RAM support. The price is $3750 to $4250.

REFERENCES

Fortran for business people by Didday, Page, Hayen, and Wayne (West Publ. Co., NY, 1978) [See ACM Computing Reviews, July 1979] features problems and examples related to payroll, sales, balance sheets, etc. An attribute is "the subtle, conversational style of the writing". Each chapter begins by explicitly stating the chapter objectives.

CODASYL Operating System Command Language JOD. This is a preliminary document for comment and review. The aim is to develop requirements for a computer-independent operating system command and response language (OSCRL). It is expected that ANSI X3H1 will use this document as a basis for developing an OSCRL standard. Send $6.00, payable to the Receiver General for Canada, to: Materiel Data Management Branch, Dept of Supply and Services, 4th floor Core B1, Place du Portage Phase III, 11 Laurier St, Hull, Quebec, Canada K1a 0S5.

Functions for Manipulating Floating Point Numbers by John Reid (Harwell, England) appeared in the December 1979 issue of ACM SIGNUM Newsletter, pages 11 - 14. For Fortran use, this paper proposes that the names EPSLN, INTXP, and SETXP be used when possible.

Note. A correspondent points out that "there is an interesting article on recursion in Fortran, in Creative Computing, Jan. 1980, pp 98 - 102".

CONCERNING FOR-WORD

This Newsletter is prepared for the U.S. Dept. of Energy under Contract W-7405-ENG-48.

The editor's name and address appear in the Mailing area on the first page of each issue. Requests for additions or corrections to the mailing list should be directed to the editor.

Correspondence on all Fortran-related topics is welcomed. Especially solicited are reviews of recent Fortran textbooks, software products, literature, etc.

X3J3 Adopts Significant Blanks Proposal

ANSI Fortran Standards Committee X3J3, at its March meeting in San Francisco, voted 19 to 6 in favor of making blanks significant in Fortran program source (effective in the next Standard).

The following proposal was adopted:

"The special character, blank, is a meaningful separator. Constants, symbolic names, statement labels, and keywords must be separated from each other by at least one operator or special character. Blank characters must not appear within the basic syntactic items of the Fortran language. The 'basic syntactic items' to which this applies are constants (other than character constants), symbolic names, keywords, and operators of more than one character. Except within these basic syntactic items, or within the datum strings of character constants or H or apostrophe edit descriptors, blanks may appear anywhere within a program unit. Blank characters count as characters in the limit of total characters allowed in any one statement. The keyword phrases GO TO, END IF, ELSE IF, and END FILE may appear with or without the embedded blank".

Further study of means for insuring coexistence of source programs with and without significant blanks is under way. One possibility is that the new program form adopted previously (see For-Word, December 1979) will expect blanks to be significant "by default", while the old program form will assume non-significant blanks. A more serious question is whether the syntax of new statements to be added to the language should use blanks (instead of parentheses, for example) as significant delimiters. This would result in much "cleaner" syntax in many instances, but would complicate the coexistence problem. Furthermore, if such statements were added it would be hard to resist "cleaning up" old statements by, for example, (at least optionally) removing the parentheses around the list in a PARAMETER statement, or perhaps even around the logical expression in an IF statement.

October Meeting in Amsterdam

The next meeting of the "Fortran Experts Group" (under the Programming Languages Subcommittee of the International Organization for Standardization, ISO) will be held in Amsterdam during the week of October 20, 1980. Any person who wishes to attend this meeting should contact his National Standards Representative to ISO. Residents of the U.S. should apply to Jeanne Adams at NCAR, Boulder CO 80307.

Fortran Presentation at IFIP Congress 80

A paper, "Fortran for the 1980's", by Walt Brainerd and Jeanne Adams, will be presented by Walt at the October 1980 meeting of the International Federation of Information Processing Societies (IFIP) in Melbourne, Australia. The paper discusses language architecture and many of the new features that will probably be included in the next Fortran Standard.

ISO Adopts Fortran 77 Standard

The International Organization for Standardization (ISO) has announced the final adoption of a new Fortran Standard, effective 1 March 1980. This Standard, ISO 1539, is technically identical to ANSI X3.9-1978 (Fortran 77).

Copies of the ISO Standard are available; however, For-Word's advice is not to purchase a copy but to buy the ANSI Standard instead. The ISO Standard is 3 pages long and consists of a cover sheet, a one-page Foreword, and a reference to ANSI X3.9-1978. The price is $6.30, or just over two dollars per page.

Loren P. Meissner
CSAM - 50B
Lawrence Berkeley Laboratory
Berkeley CA 94720

FIRST CLASS

PUB 345

PUB-345 [6:2 6/80] 2300

Other X3J3 Actions

At its May meeting in Aberdeen, Maryland, X3J3 adopted a number of new proposals. **Readers are reminded that X3J3 actions may be rescinded or extensively revised before final standardization.**

1. Variable-length Character Type

This feature permits variable-length character strings within a declared maximum length. The current length is set implicitly during assignment (or input, etc.), and is available via the LEN intrinsic function. Fixed-length character strings are also retained. Substrings and constants are always considered fixed-length.

Variable-length and fixed-length strings can be mixed in most contexts, except that corresponding actual and dummy arguments must be either both fixed or both variable.

2. Character Intrinsic Functions

Five new intrinsic functions for character handling were adopted. Others are under consideration.

VERIFY has two argument strings, and returns the position of the first character in the second string that does not appear anywhere in the first string.

TRIM deletes trailing blanks.

ADJUSTL shifts leading blanks so that they become trailing blanks; ADJUSTR does the opposite.

REPEAT generates a sequence of concatenated copies of a given string.

3. Global COMMON Declaration

A statement similar to COMMON defines a named collection of global data elements, but with no implied storage association or sequence properties. Once such a definition has appeared, other program units can "import" the data by simply declaring the name of the collection.

4. Generalized Precision Attribute

A first step was taken toward adoption of a generalized REAL (and COMPLEX) data type. It was decided that there will be a "default" precision, and that the declaration for a variable with the default precision should be identical to the Fortran 77 REAL (or COMPLEX) declaration.

Proposals to further refine this feature are in preparation.

5. Whole Array Sections

An extension to the Array Processing features (see For-Word, September 1979) was adopted. This feature permits the values of certain array subscripts to vary over their declared range, while others remain fixed. (Still under study are proposals which will permit some or all of the subscripts to vary over a portion of their declared range.)

An array name is followed by a parenthesized list, similar to the list of subscript expressions in an array element identifier. Certain positions in this list may contain subscript expressions as usual, while others may contain the "section selector" symbol (probably either * or :). For each position where an expression appears, the subscript value is fixed and the number of dimensions of the resulting "array object" is reduced by one (from the number of dimensions of the original declared array). For each position where the selector symbol appears, the number of dimensions is not reduced and the "array object" maintains the entire range of subscript values declared for that dimension. Thus the number of dimensions in the array object is equal to the number of positions that contain the selector symbol, and the upper and lower bounds for each of these dimensions are those originally declared for the array name.

Such an "array section" object may be used in any context where the array name may be used alone: In an array assignment statement, in an input or output list, as an actual argument, etc.

6. Language Architecture

Rules were adopted relating to the inclusion of features in the Obsolete Features Module and in the Language Extension Module(s). Each such module is to be an integral part of the Fortran Standard. A feature deleted from the Core or from a Language Extension Module will be placed in the Obsolete Features Module for at least one full revision cycle of the Standard. Obsolete features must be compatible with features in the Core or Language Extension Modules, unless some form of "option statement" is provided. Language Extension Modules may be added, but not revised or deleted, between revisions of the Core Standard.

Interface Mechanisms Considered

X3J3 is also working on methods for interfacing modules (particularly Language Extension Modules and Application Area Support Modules) with Core Fortran.

It is considered likely that some modules will require non-Standard syntax or semantics (such as the CODASYL Data Base "INVOKE" statement), so that some form of "compiler escape" will have to be provided. An important question is whether all such modules must be described in such a way that non-Standard statements can (theoretically, at least) be translated to Standard syntax and further processed by the "regular" Fortran compiler.

Interface mechanisms being considered include an enhanced CALL statement, a MACRO processing facility, and a USING statement that would invoke a specified preprocessor.

X3J3 Proposal Processing Under Way

A large number of proposals are to be considered by X3J3 at the next few meetings. The plan is that all major proposals should be presented in the near future, so that the Committee can get a broad view of what is likely to be included in the next Standard. It is understood that major adjustment will then be necessary to merge all of these features into a coherent language. Another major task will be to determine which features belong in Core Fortran, and which are to be part of a Language Extension Module or an Application Area Support Module.

Forthcoming meetings are scheduled for August 1980 in Aspen CO; October 1980 in Fort Lauderdale FL; January 1981 in Berkeley CA; March 1981 in Austin TX; May 1981 in Toronto, Ontario; and July 1981 in Santa Fe NM.

Meetings are open to the public, but facilities are limited. Persons who would like to attend a meeting as Observers should notify Martin Greenfield at MS 844a, Honeywell Information Systems, 300 Concord Rd, Billerica MA 01821.

A Note of Appreciation

Thanks to all who took time to respond to the For-Word questionnaire on significant blanks, and to others who wrote letters expressing concerns on this issue.

Preprocessor with Recursion?

If you are aware of a Fortran preprocessor that allows recursion, please notify John M Hosack, Math Dept, New Mexico Highlands University, Las Vegas NM 87701.

GUEST ARTICLE

Fortran Increasing in Significance
(by General W Roldjer)

Following its heroic action in making blanks significant to aid syntactic interpretation, the ANTSY Fortran Committee JX3.3 is pioneering an approach to aid semantic interpretation. As a first step, an ASSERT statement has been proposed, which can aid the compiler in making decisions during optimization. Such an assertion takes precedence over the actual code in the event of a conflict, because it represents what the programmer really wants.

A proposal to add the READ MIND statement to Fortran was favored on a straw vote, but after extensive discussion was rejected by the Committee, on the grounds that the Fortran Standard should not specify the medium from which the processor obtains the source program. If a MACRO facility is adopted, however, this statement may be reconsidered at that time.

In a more general attack on the problem of semantic significance, the ANTSY committee is considering making comments significant. This breakthrough in programming style recognizes that comments are the most reliable indication of the programmer's true intent. While the Fortran code, and even ASSERT statements, might contain bugs or other errors, comments can be depended upon to represent the desired semantics of the algorithm.

As proposed, comments following a statement on the same line (as permitted with the new source form adopted in 1979) would affect only the semantics of that line, and would have the highest semantic priority. A block of one or more contiguous comment lines would apply to the semantics of all subsequent statements until the next block of comment lines appears (unless superseded for one or more individual statements by end-of-line comments). A statement that conflicts with any comment that applies to it would be indented beyond the right margin of the source program listing, and a non-conflicting Fortran statement would automatically be generated by the compiler to replace it.

Once significant comments are available, the ASSERT statement would be redundant, and therefore would be relegated to the Obsolete Features Module.

In yet another related action, JX3.3 took the first steps toward eliminating independent compilation of program units by adopting a "Global Common" block. This line of development is expected to lead eventually to "significant archives", which will contain all Fortran program statements ever written. Thus any program, or any portion of a program, once written would always thereafter be available for use. Since no program statement would ever have to be written again, once it had been stored in the archives, programming effort would be reduced dramatically. Eventually, it is to be expected that any statement that is needed could be found somewhere in the archives, and thus programming per se would be eliminated entirely.

Editor's Note: Responses to the above Guest Article should be directed to the file described in X3.9-1978, page B-7, lines 37 - 38.

ANNOUNCEMENTS

Standard Syntactic Metalanguage

A working party of the British Standards Institution has proposed a Standard syntactic metalanguage, based on Naur's report (Computer Journal, January 1963). A draft of the proposal can be obtained from Roger S Scowen (Computing Services Unit, National Physical Laboratory, Teddington Middlesex TW11 0LW, England). An appendix gives, as an example, the syntax for 1966 Standard Fortran expressed in the proposed metalanguage.

Fortran 77 Compiler from Harris

Harris Computer Systems Division has implemented a full Fortran 77 compiler. Extensions include 63-character names, additional control structures (FOR, LOOP, WHILE, and DO-UNTIL), in-line assembly language, conditional compilation, asynchronous (BUFFER IN/OUT), and ISA process control and bit manipulation features. Further information is available from Lyle A Plitt, Harris Computer Systems Division, 2101 W Cypress Creek Rd, Ft Lauderdale FL 33309, or phone (305) 974-1700.

Fortran 77 Compiler from ACT Corp

Advanced Computer Techniques Corporation announces a full Fortran 77 compiler written in PASCAL. The first version is now running on Honeywell Level 6 machines. Structure of the compiler permits replacement of the code generation routines (for a new hardware architecture) with minimum effort. The basic price is $125K, including documentation and training. Further information is available from L Hersh, ACT, 437 Madison Ave, NY 10022, or phone (212) 421-4688.

Decision Table Converter

Integrated Business Systems has a preprocessor that converts horizontal, extended-entry decision tables to Fortran source form. In the preprocessor input, the number of columns and the width of each column are set by the user.

A simple report writer is also available.

Further information, including a short bulletin with several examples, or more complete product specifications, is available from Daniel F Langenwalter, Integrated Business Systems, 26777 Lorain Rd - Suite 213, North Olmsted OH 44070.

IMSL Minimal Test Package

A series of Fortran programs, designed to test each subroutine in the IMSL Library for numerical accuracy and proper operation, is now available. The package, consisting of some 25,000 Fortran source lines, is particularly intended for users of non-Standard or altered compilers. It is available to IMSL subscribers for only $250 per year.

International Mathematical and Statistical Library, 6th Floor, NBC Bldg, 7500 Bellaire Blvd, Houston TX 77036 (713) 772-1927.

Boeing Graphics Package

Boeing Computer Services Co announces DIGRAF ("Device-independent Graphics from Fortran"), consisting of more than 200 subroutines for generating and interacting with graphics images. It does not include applications-oriented functions such as graphing, contouring, etc; rather, it is intended as a base for developing systems of applications.

The package is based on standardization efforts under way in ANSI (X3H3), ACM-SIGGRAPH (Graphics Standards Planning Committee) Core Graphics System, and international graphics activities under ISO.

Further information is available from Graphics Consulting Service, MS 9C-01, Boeing Computer Services Co, PO Box 24346, Seattle WA 98124 (206) 575-5012.

CONCERNING FOR-WORD

This Newsletter is prepared for the U.S. Dept. of Energy under Contract W-7405-ENG-48.

The editor's name and address appear in the Mailing area on the first page of each issue. Requests for additions or corrections to the mailing list should be directed to the editor.

Correspondence on all Fortran-related topics is welcomed. Especially solicited are reviews of recent Fortran textbooks, software products, literature, etc.