AAEC/E454

AAEC/E454

4

AUSTRALIAN ATOMIC ENERGY COMMISSION
RESEARCH ESTABLISHMENT
LUCAS HEIGHTS

COMFORT - AN INTERACTIVE FORTRAN SYSTEM FOR THE IBM360 COMPUTER

by

B.E. CLANCY

December 1978

ISBN 0 642 59659 X

AUSTRALIAN ATOMIC ENERGY COMMISSION
RESEARCH ESTABLISHMENT
LUCAS HEIGHTS

COMFORT - AN INTERACTIVE FORTRAN SYSTEM FOR THE IBM360 COMPUTER

by

B.E. CLANCY

ABSTRACT

COMFORT is an interactive programming system for the IBM360 computer which allows a user to edit, save, compile, execute and monitor small to medium size FORTRAN programs from a terminal. The system includes a powerful graphics package.

This report describes the system and gives instruction for its use at the Australian Atomic Energy Commission Research Establishment.

National Library of Australia card number and ISBN 0 642 59659 X

CONTENTS

1. INTRODUCTION

The advantages of carrying out certain computing tasks in an interactive fashion are well known. This report describes an interactive FORTRAN program which has been written for the IBM360/370 series of computers and which includes a simple but powerful graphics package. The program interacts with only one user, and requires less than 240 K bytes of core storage. At the AAEC Research Establishment, communication between the central computer and the interacting user is handled by the AAEC Dataway [Richardson 1969] but the system could be applicable to other computer networks.

2. STRUCTURE OF THE COMFORT PROGRAM

In this section, a brief description is given of the structure of COMFORT - how the user interacts with it and how the program responds to the user's requests.

At the heart of the program is a control segment, namely the subroutine CONTRL. This segment supervises the user's interaction with the program and calls in other segments as they are required. Three other segments can be identified: an EDIT segment which allows the user to define and modify the FORTRAN program and some of the associated data; a COMPILE segment which uses the FOREX compiler of Robinson [1968, 1977] to convert the user's program to machine code and provide linkage to library subprograms; and an EXECUTE segment which passes control to the user's program, but monitors it for errors.

Much of the information supplied by the user during a session is saved on two card-image data sets. The first of these contains the card images defining the user's FORTRAN program and this data set is read by the compiler while the COMPILE segment is in control. The second data set - called herein the user's SYSIN data set - is logically FORTRAN unit 1 and can be read by the user's program. This data set thus corresponds to those data which the user would normally supply in the input stream of a batch submitted job.

2.1 COMPILE Segment

The Robinson FOREX compiler comprehends a dialect of the FORTRAN IV language. Programming must conform to the following restrictions:

- The only variable types allowed are single precision (4 byte) integers or reals.

- Variable types are determined by the first letter of their names. No explicit or implicit retyping is allowed.
- Only the simplest LOGICAL IF statement is recognised so that the logical operators .AND. .OR. .NOT. are not understood.
- The ARITHMETIC IF statement is supported.
- Mixed mode arithmetic is not allowed.
- Direct access input/output statements in the IBM FORTRAN style are not supported.

The user's program has some other restrictions: the blank common area cannot exceed 1000 words; the user's source program must be less than 900 lines; the compiled program must occupy less than 70 K bytes, and of this space dimensioned variables must occupy less than 24 K bytes.

Linkage to FORTRAN library subprograms is generated explicitly in the COMPILE segment from tables of names and addresses set up when COMFORT itself was compiled and linked.

Subprograms referred to by the user's main program must therefore:

- be a library subroutine or function,
- have their FORTRAN source given as part of the user's program, or
- be accessible to the LINK subprogram described below.

A list of the library programs is given in Section 4.

2.2 EDIT Segment

When activated, this segment brings into core the user's source program and SYSIN data and allows the user to modify, delete or add any lines to them, the actual editing being performed by a subroutine named EDIT. This is a FORTRAN version of the ØLOGON program of Backstrom [1976]. There would be little difficulty, for another computing environment, in inserting any other editing subroutine in place of the present EDIT subroutine.

2.3 EXECUTE Segment

This segment consists of the compiled user's program and executes under the supervision of the ØSPEX error monitor of Cawley [1977]. This monitor traps errors detected while the user's program is executing, writes appropriate messages to the user's terminal and either permits the program to continue after a standard 'fix up' or returns to the control segment. This return to the control segment occurs if the errors are serious (those for which the normal IBM FORTRAN monitor terminates the user's job step), or if minor errors have occurred an excessive number of times. The return to the control segment allows the user to alter the program before trying again to execute it.

2.4 Control Segment

Unless control has been passed to one of the segments described earlier, this segment supervises the user's interaction with the computer. CONTRL requests instructions from the user, checks them for validity and, if possible, carries them out before requesting further instructions. When a session at the terminal begins, the user is first identified and the program then asks whether or not that terminal is one which supports graphics. CONTRL is then ready for an instruction, which it signals by typing the line ENTER COMMAND at the terminal and waiting for a response.

The user should respond by typing a command (and, in some cases, a command followed by an additional word). The response is transmitted to the program when the user presses 'carriage return' at the terminal. The final user response is to transmit the command STOP which closes down the terminal and ends execution of the COMFORT program.

The most common additional word transmitted with a command is called a 'jobname'. This is a string of three to eight characters which will be associated with the user's FORTRAN program when control is passed to one of the other segments. As will be seen later, it is possible for the user to save programs (and data) on a library data set during one session with COMFORT, and to recall this program at the same or a later session. The jobname is used as an identifier of such a saved program.

3. DESCRIPTION OF COMMANDS

In this section, the commands understood by the control segment will be listed and their functions described. Some commands have aliases - often abbreviations of the full keyword - and these too are listed. Only the first four non-blank characters of a command are in fact significant and in the following subsections these are underlined. Compliance with some of the commands requires the user to send additional information to the control segment when necessary.

3.1 Tracing Commands

TRACE sets printer trace on

NOTRACE sets printer trace off

If the trace is set on, then all commands detected by the control segment will be logged on the user's printed output together with details of programs and data saved or loaded from the COMFORT library. The default status is to set the trace on.

3.2 Jobname Commands

The command WHO (alias ?AMI) causes the control segment to send to the terminal the jobname at present associated with the job. The command IAM followed by a 3- to 8- character string instructs the control segment to use the string as a new jobname, provided that the string begins with the user's initials.

3.3 Library Processing Commands

These commands are

SAVE	alias	SA
LOAD	alias	LO
DELETE	alias	DE
ADD	alias	A
?SAV	alias	J

Nominally each command should be followed by a jobname but, if none is transmitted with the command, the user's present jobname is used in default.

SAVE stores the existing user's program, SYSIN data and the values of all COMMON variables as members of a partitioned data set (named LIBRARY). An identifying comment card is inserted at the front of the user's program. This has the date and time at which the member was saved. If a member already exists with the given name, the user is asked to verify his request (by a simple Yes/No response) before replacing the old member with the new one.

LOAD is the reverse of SAVE. The user's existing program, data and common values are replaced by those of the named library member.

DELETE removes from the library the member with the nominated jobname provided that the user verifies this request.

ADD adds to the existing user's program, data and common values, the program and data lines previously saved under the nominated jobname. If the user has constructed coding for a frequently used subprogram, it may be convenient to save this coding as a member and add it to other members when necessary.

?SAVE produces at the terminal a list of those member names in the library which begin with the three initials of the nominated jobname.

Thus ?SAVE on its own produces a list of those member names beginning with the user's own initials. If instead of a valid jobname the user appends the character * (*i.e.* ?SAVE *,), then all members of the library are listed.

Although a user may load (or add) any member from the library, only members whose names begin with the user's initials can be deleted or saved.

3.4 User's Program Commands

These commands are associated with the FORTRAN coding which defines the user's program.

EDIT passes control to the Edit segment which allows the user to modify or add to the coding defining his program as well as to the lines (notionally card images) which will comprise the SYSIN data. At the beginning of a session at the terminal, no program will have been defined and the user can enter this command to begin writing the FORTRAN coding. Control stays in the Edit segment until one of the control commands is transmitted to the segment, at which stage CONTRL resumes its supervision. Use of the interactive edit procedure is the same as that described fully by Backstrom [1976] and partly by Clancy [1977].

PF is a command which causes a listing of the user's existing program to appear in the printed output from the COMFORT session.

PD produces a corresponding listing of the contents of the user's SYSIN data set.

?F is a command which lists, at the terminal, the most recently compiled version of the user's program. 'Printed' output from the compiler is actually written onto a COMFORT system data set and when the ?F command is sensed, the control segment simply copies this, one line at a time, to the user's terminal. If the compiler has detected errors, an appropriate message will be present in the compiler output. After correcting some of these with the edit facility, the user may use ?F to re-examine the compiler output so as to ensure that all errors have been corrected before invoking the compiler again.

3.5 'COMMON' Commands

Although the FOREX compiler recognises only a single blank COMMON area of 1000 words this area has, within the COMFORT system, one attribute of a labelled COMMON area. The user may, before execution of the program, define values of any common variable and also interrogate the system to find what values have been assigned to the variables.

COMMON allows the user to assign real or integer values to the variables whereas

?COMMON causes a list of the non-zero variables to be transmitted to the terminal.

PCOMMON causes a compressed list of the common variables to appear in the printed output for the COMFORT session.

After receiving the command COMMON, the control segment replies with a request ENTER COMMON VARIABLES and is ready to receive the responses. Each positive response has three components - a letter I or R to indicate an integer or real variable value, an address number in the range 1-1000 to indicate the position in COMMON which is to be initialised and, finally, the value to be inserted. Thus, to set the third COMMON location to be an integer 5 and the fourth COMMON location to be a real -3.0, the user's responses would be

```
I    3    5
R    4    -3.0
```

A blank line (or immediate carriage return) sent as a response signals that no more values are to be initialised.

3.6 Execute Command

GO (alias ST or S) is the command which initiates execution of the user's program under the control of COMFORT. If the user's program has not already been compiled successfully, control is first given to the compile segment (*cf* Section 2.1). If FORTRAN errors are detected, the compiler output is sent to the user's terminal just as if the command ?F (Section 3.4) had been issued. The user may then EDIT to correct the FORTRAN coding and try again. If no compilation errors are detected, control is passed to the EXECUTE segment which branches to the beginning of the user's compiled program. If the user's program has been compiled successfully before - and execution is just being repeated, perhaps with different data - the compile step is bypassed.

The user's program continues executing until

(i) a FORTRAN STOP or END is encountered,

(ii) the error monitor ØSPEX decides to terminate the user's program, or

(iii) a specified programmable interrupt occurs.

In the first two cases, a return is made to the control segment; in the third case the user may elect to stop the program or allow execution to continue, possibly with some modification of the common variables.

Programmable interrupts are described more fully in Sections 4.3 and 4.4.

3.7 Information Commands

The command FOREX is issued when the user wishes to find out information about the compiler. After issuing the command, the user must select one of four options which in turn produce at the terminal:

(i) a list of subroutines available,

(ii) a list of function subprograms,

(iii) a list of 'macro' subprograms, and

(iv) a list of the special features which distinguish the FOREX dialect from the A.N.S standard FORTRAN language.

The command TIME (alias GOTIME) produces as a response from the control segment a statement of how much time is left before the session must stop. Two times are relevant here:

(i) the number of central processor unit (CPU) seconds remaining before the IBM360/370 will terminate the job, and

(ii) the number of real (wall-clock) minutes remaining before the COMFORT program will terminate the user's session. The CPU time limit is determined by the user's job card but the wall clock restriction is quite independent of this. COMFORT will terminate the user's session after 30 minutes during the 8 a.m. - 6 p.m. period.

The command HELP (alias ?) produces at the user's terminal a list of the recognised commands with a brief statement of their functions.

The query ? can also be used while the commands FOREX and COMMON are being processed. When the control segment asks for additional information, a list of instructions will be sent to the user if the user's response was to type the query.

3.8 Ending Interaction

The command SUSPEND can be issued if the user wishes to suspend the session with COMFORT for a short period and then resume at the same or another terminal. Unless a number is appended to the command, the user will be asked to specify the number of minutes of suspense required, up to a maximum of ten. The user's existing program and data are saved as a special member of the library, interaction with the terminal ceases and COMFORT goes into a wait state for the nominated period. At the end of this period, the user should log on at some terminal and resume interaction. Five minutes' grace is allowed for the user to get access to a terminal before the COMFORT job is terminated. The command STOP closes down the terminal and terminates the job immediately.

3.9 Miscellaneous Output Commands

The command P acts as a combination of the commands PF and PCOMMON. It causes a listing of the user's program, SYSIN data and common values to appear in the printed output of the COMFORT session.

The command PUNCH causes the user's source program and SYSIN data to be produced as punched output of the session.

The command PRINT ANY MESSAGE causes the string ANY MESSAGE to appear in the printed output of the session. This can be useful in separating blocks of printed output produced by successive executions of the user's FOREX program.

3.10 Job Submission Commands

The two commands described here cause a batch job to be submitted to the IBM360 *via* the internal reader. After sensing the command, the control segment requests from the user such additional information as is necessary to construct valid job and JCL cards before submitting the job.

The command LOGON is used to transfer the user's program and SYSIN data into the ØLOGON library [Backstrom 1976].

SUBMIT (alias SUB or SU) is the command to submit the user's program and data as an ordinary batch job.

4. FOREX SUBPROGRAMS

As well as the subprograms for which the user supplies his own FORTRAN coding, a library of subroutines, functions and FOREX macros is available to the user. When interacting with the program, the user can obtain lists of these by issuing the command FOREX. The resulting lists are given below.

4.1 Function Subprograms

Library functions are

```
FIXPI#    FRXPI#    FRXPR#    ALINK     ALOG
ALOG10    AMAX0     AMAX1     AMIN0     AMIN1
ARCOS     ARSIN     ATAN      ATAN2     COS
COSH      COTAN     EI        ERF       ERFC
EXP       EXPI      EXPRNF    FLOAT     FLTRNF
FTTGO     GAMMA     ITTGO     LINK      MAX0
```

MAX1	MIN0	MIN1	RAND	RNMAXF
SFLRAF	SIN	SINH	SQRT	TAN
TANH				

Except for the two entries underlined, the functions listed are members of the AAEC system libraries SYS1.FORTLIB, AAE.FORTLIB or PHYS.FORTLIB; the appropriate reference manuals should be consulted for further information. Functions ALINK and LINK are described in Section 6.

4.2 Subroutine Subprograms

Library subroutines are

IBCOM#	AZIRN	BES	CNSCAN	DRECT
DWCAN	DWCASE	DWCTLR	DWCTLW	DWHOME
DWREAD	DWREAS	DWRITE	DWRITN	EMPTY
EXEM	EXPI	FORTEF	FORTRD	FORTRW
FORTSF	FORTWR	GTEND	GTISO	INREAD
LIBER	LINK	MOVE	MOVECH	OPSCAN
PDUMP	PJAIMP	PJCAMP	PJDRAW	PJEND
PJFITS	PJFRAM	PJGRID	PJHEAD	PJHIDE
PJPAPE	POLRN	QUIKWA	RTARC	RTCURV
RTFAKE	RTPAPE	RTPLOT	RTPRJW	RTPROJ
RTPURV	RTRAP	RTRLIM	RTTDEG	SCAN
SCANL	SCANS	SEND	SETBUF	SID
SMVMPY	SORT	SPMASK	SSMMPY	STSCAN
SWAIT	XYAXES	XYBIRO	XYBNDS	XYBOX
XYBUF	XYBUFN	XYCONS	XYCURS	XYCURV
XYEND	XYERRB	XYFAKE	XYFRAN	XYHEAD
XYHIST	XYHMAX	XYLEG	XYLEGH	XYLEGI
XYLEGO	XYLEGS	XYLEGW	XYLINE	XYLSEG
XYNAMX	XYNAMY	XYNUMB	XYPAPE	XYPESC
XYPEST	XYPEXT	XYPLOT	XYPLTF	XYPRJW
XYPROJ	XYPNTS	XYPSEG	XYPUMB	XYREST
XYSETL	XYSETS	XYSETT	XYSHFC	XYSKEW
XYSPCE	XYSSZE	XYSUSP	XYTEXT	XYVMAX
XYVPOS	XYWAY	XYXAXS	XYXFMT	XYYAXS
XYYFMT				

Many of these subroutines are descibed in the reference manuals for the AAEC system libraries mentioned in Section 4.1. The subroutines with names beginning with DW are part of the DWCOMM package of Cawley & Trimble [1977] and those having names beginning with PJ, RT or XY are part of the XYPLOT graphics package [Trimble 1978]. The subroutine LINK is described in Section 6.

4.3 Library Macros

The macro statement recognised by FOREX is an addition to the FORTRAN language. In the COMFORT system, a macro takes the form of a command word followed in the same FORTRAN statement by a character string 'MESSAGE'. The library macro command words are as follows:

STOP MESSAGE is typed at the terminal and COMFORT returns to the control segment.

TYPE MESSAGE is typed at the terminal and the user's program continues execution.

PRINT MESSAGE is written as part of the job printed output and the program continues.

GOTIME MESSAGE and timing information is sent to the user's terminal.

PAUSE MESSAGE appears at the terminal; the user's program halts and requests instructions from the terminal by typing the words ENTER OPTION. An immediate carriage return from the terminal lets the program continue. A query ? response from the user produces a list of the options which are a selection of the control segment commands. The option STOP here simply returns to the control segment.

INTERRUPT usually has no effect. However, if the user has pressed the query ? at the terminal, an interrupt flag is set and the INTERRUPT acts like a pause macro. As soon as the program pauses, the flag is reset and the interrupt will not be effective until the query is pressed again.

4.4 The SCAN Interrupt

A restricted version of the free input subroutine SCAN [Bennett & Pollard 1967] is available with COMFORT. In this version, all SCAN errors during execution of the user's program are trapped and the program returns to the control segment.

If the first column of any record read by SCAN should contain a query ?, execution is interrupted and the user has the options described in the INTERRUPT macro. When the user's program resumes execution, the remainder of the record with the query is ignored. This SCAN interrupt works no matter what logical FORTRAN unit is being read from (including the user's terminal).

5. LOGICAL FORTRAN UNITS FOR INPUT-OUTPUT

If, when submitting the job, the user supplies DD cards for FORTRAN logical units 10,11,... *etc*., reference may be made to these units in the program in whatever way is required. Units 1 to 9 are, however, reserved for special uses in the COMFORT GO step.

Unit 1 is a card image data set accessible to the user's compiled program by standard FORTRAN input/output statements. Each time the EDIT segment gains control, the contents of this data set are brought into core together with the FORTRAN source of the user's program and both can be changed by the editor. When in core, the two sets of card images are separated by a Job Control Language (JCL) statement, //GO.SYSIN DD *.
On completion of the edit phase, the (possibly) changed card images are copied back onto the data sets.

Unit 2 is the logical unit for punched card output.

Unit 3 is the logical unit for printer output.

Unit 4 is reserved for system use.

Units 5,6 are assigned to the user's terminal for FORTRAN reads and writes respectively.

Unit 7 is reserved. The user's FORTRAN coding is written there for the FOREX compiler to read.

Unit 8 is reserved for printed output from the compiler.

Unit 9 is reserved for system use.

6. ACCESS TO EXTERNAL FUNCTIONS AND SUBROUTINES

With the FOREX compiler, linkage between the user's program and the subprograms listed in Section 4 is set up directly at compilation time. COMFORT contains a list of the available subprogram names with their entry points and these are passed to the compiler. To give the user's program complete access to additional subprograms, it would be necessary to add them to the list, recompile COMFORT and link the new program. This procedure is too clumsy for general use and yet there are occasions when access to additional routines is necessary.

A partial solution to this conflict has been provided by including, as part of the COMFORT system, an additional subprogram LINK which may be called as a subroutine,

```
        CALL LINK (argument list)
```

as a real function,

```
        ANSWER = ALINK (argument list)
```

or as an integer function,

```
        NUMBER = LINK (argument list)   .
```

In each case, the first item in the argument list must be the name (as an eight-character string) of the subprogram to which access is wanted. The remaining items of the argument list are the variables which would form the full argument list if the subprogram were being accessed in the normal fashion. Thus to get the effect of the normal statements

```
    (i)  Y1  =  ATAN2 (A,B)
   (ii)  CALL SUBA(X,Y,Y1)   ,
```

the user could code the corresponding statements

```
    (i)  Y1 = ALINK('ATAN2   ',A,B)
   (ii)  CALL LINK('SUBA    ',X,Y,Y1)  .
```

The subprogram LINK makes use of the IBM operating system supervisor macro instruction 'LINK'. When this is called, a search is made of all data sets concatenated with the STEPLIB data set until a load module with the required name is encountered, the names being ATAN2 and SUBA in the preceding example. When the module is located, the supervisor loads this into an unused core area and passes control to it so as to carry out the subprogram task. Return to the user's program is *via* the LINK routine which relinquishes the core area for later use.

Some preplanning by the user is required if the LINK feature is to be used. The COMFORT steps (or catalogued procedure) must be preceded by a step which invokes the linkage editor and writes the necessary linked load modules as members of a partitioned data set. This data set must also be concatenated with the COMFORT data set GO.STEPLIB.

With two exceptions, the user's linkage editor steps should include sufficient routines to resolve any external references which the requested subprogram may call. The steps should also specify an entry point to the module and should give the module a specific name. The two external references which must be left unresolved are those to the input/output routine IBCOM# and to the error monitor IHCERRM. As a consequence, the linked subprogram must neither perform any input/output nor produce any errors. An example of a job using this facility is given in Appendix A.

7. EXECUTION TIME ERRORS IN THE USER'S PROGRAM

The ∅SPEX routine of Cawley [1977] supervises the execution of the user's compiled program. Errors are trapped and appropriate error messages transmitted to the user's terminal, before allowing the program to continue execution, after a standard fix up is taken. If no standard fix up is possible, or if an excessive number of errors has been detected, the user's program is terminated and the error monitor transfers to the control segment so that the user's FORTRAN can be corrected before the program is re-executed.

If the abnormal termination occurs during execution of a subprogram brought into core by the LINK macro, the space occupied by the subprogram will not be freed for later use by other segments of the COMFORT program. Continued execution of the program will then often result in a catastrophic failure of the COMFORT program and the user's session at the terminal will end dramatically. For this reason, the LINK routine should be used with great care.

With this exception, the COMFORT system is believed to be foolproof - once the user starts a session then it should be able to continue without being terminated because of errors.

8. ACKNOWLEDGEMENTS

Thanks are due to G.S. Robinson for modifications to his FOREX compiler and advice on its use, to G.D. Trimble for assistance with the error trapping routines and for advice on the use of the DWCOMM package, and to R.J. Cawley for making available his ∅SPEX routine. J.P. Pollard acted as a willing trial user of early versions of COMFORT and provided a number of valuable suggestions for its improvement. It is a pleasure to acknowledge their assistance.

9. REFERENCES

1. Backstrom, R.P. [1976] - ∅LOGON - An Interactive Job Submission Facility for the IBM360 Computer. AAEC unpublished report.

2. Bennett, M.W. & Pollard, J.P. [1967] - AAEC/TM399.

3. Cawley, R.J. [1977] - AAEC private communication.

4. Cawley, R.J. & Trimble, G.D. [1977] - AAEC/E425.

5. Clancy, B.E. [1977] - AAEC/E408.

6. Ellis, P.J. [1970] - AAEC/E206.

7. Richardson, D.J. [1969] - *Aust. Computer J.*, Vol.1, No.5. Additional information given in references 4, 6 and 10.

8. Robinson, G.S. [1968] - AAEC/E190.

9. Robinson, G.S. [1977] - Modifications to the FOREX Compiler. AAEC unpublished report.

10. Trimble, G.D. [1978] - AAEC/E437.

APPENDIX A

THE USE OF LINK SUBPROGRAMS

An example of the use of LINK subprograms as part of a COMFORT job is shown in Figures A1 to A4. For this example, the user wishes to access three subprograms not available as part of the COMFORT system, *viz*

a subroutine	DATE	,
an integer-function	MOD	, and
a real function	AMOD	.

Figure A1 is a listing of the job deck submitted to the IBM360 and Figure A2 is the operating system output for the job. The first step (named LINK) invokes the linkage editor to create separate load modules for the three subprograms and these are passed *via* the temporary data set &&AAEC to the COMFORT system.

Figure A3 is a listing of the output provided by the linkage editor and Figure A4 shows the printed output produced by COMFORT. Included in this last listing is the user's program which needs the three LINKed subprograms.

FIGURE A1

```
//BECOMFEX JOB ('********/DOOPHICI',N1),B.E.CLANCY,
//       MSGLEVEL=(2,0),CLASS=I,TIME=10
//*ROUTE PRINT PHYS
//*
//* INVOKE LINKAGE EDITOR TO CREATE THREE MODULES FOR USE BY COMFORT
//*
//LINK EXEC FORTHL,PARM.LKED='LIST,LET,NCAL'
 INCLUDE SYSLIB(DATE)
 ENTRY DATE
 NAME DATE(R)
 INCLUDE SYSLIB(MOD)
 ENTRY MOD
 NAME MOD(R)
 INCLUDE SYSLIB(AMOD)
 ENTRY AMOD
 NAME AMOD(R)
//*
//* NOW EXECUTE COMFORT
//*
//RUN EXEC COMFORT
//GO.STEPLIB DD
//            DD DSN=&&AAEC,DISP=(OLD,PASS)
//*
//*              DSN=&&AAEC CREATED BY LINKAGE EDITOR
//*    AND MUST BE CONCATENATED WITH    STEPLIB
//*
```

FIGURE A2

```
//BECOMFEX JOB ('********/DOOPHICI',N1),B.E.CLANCY,
//       MSGLEVEL=(2,0),CLASS=I,TIME=10
***ROUTE PRINT PHYS
***
*** INVOKE LINKAGE EDITOR TO CREATE THREE MODULES FOR USE BY COMFORT
***
//LINK EXEC FORTHL.PARM.LKED='LIST,LET,NCAL'
//SYSIN     DD *                GENERATED STATEMENT
***
*** NOW EXECUTE COMFORT
***
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /LKED    / START 78215.1406
IEF374I STEP /LKED    / STOP  78215.1406 CPU   0MIN 01.02SEC MAIN  96K LCS
*** CONDITION CODE = 000(HEX)
//RUN EXEC COMFORT
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /PLT     / START 78215.1406
IEF374I STEP /PLT     / STOP 78215.1406 CPU    0MIN 00.32SEC MAIN  14K LCS
*** CONDITION CODE = 000(HEX)
//GO.STEPLIB DD
//            DD DSN=&&AAEC,DISP=(OLD,PASS)
***
***               DSN=&&AAEC  CREATED BY LINKAGE EDITOR
***      AND MUST BE CONCATENATED WITH    STEPLIB
***
IEF142I - STEP WAS EXECUTED - COND CODE 0000
IEF373I STEP /GO      / START 78215.1406
IEF374I STEP /GO      / STOP  78215.1407 CPU   0MIN 03.05SEC MAIN 234K LCS
*** CONDITION CODE = 000(HEX)
IEF375I  JOB /BECOMFEX/ START 78215.1406
IEF376I  JOB /BECOMFEX/ STOP 78215.1407 CPU    0MIN 04.39SEC
HIGHEST CONDN CODE = 000(HEX)
```

FIGURE A3

```
 F88-LEVEL LINKAGE EDITOR OPTIONS SPECIFIED LIST,LET,NCAL
          DEFAULT OPTION(S) USED - SIZE=(90112.6144)
IEW0000      INCLUDE SYSLIB(DATE)
IEW0000      ENTRY DATE
IEW0000      NAME DATE(R)
****DATE       DOES NOT EXIST BUT HAS BEEN ADDED TO DATA SET
```

```
IEW0000      INCLUDE SYSLIB(MOD)
IEW0000      ENTRY MOD
IEW0000      NAME MOD(R)
****MOD        DOES NOT EXIST BUT HAS BEEN ADDED TO DATA SET
```

```
IEW0000      INCLUDE SYSLIB(AMOD)
IEW0000      ENTRY AMOD
IEW0000      NAME AMOD(R)
****AMOD       DOES NOT EXIST BUT HAS BEEN ADDED TO DATA SET
```

FIGURE A4

```
COMMAND TRACE IS SET ON#
JOBNAME CHANGED TO "BECOMFEX"                     ****
----COMMAND----LO  ----
BECOMFEX WAS LOADED                               ****
----COMMAND----PF  ----


C***BECOMFEX SAVED                        78.215 12.29.51 MVT
C---- STANDARD DATE SUBROUTINE IS NOT IN COMFORT
C----      BUT MAY BE ACCESSED THROUGH SUBROUTINE LINK
C
C     CALL DATE (ID)
C
C------------------------------------------------------------
C---- FOLLOWING IS CALLING SEQUENCE THROUGH SUBROUTINE LINK
      DIMENSION ID(3)
      CALL LINK('DATE    ',ID)
      WRITE(6,2)ID
      WRITE(3,2)ID
    2 FORMAT(' TODAY IS ',3A4)
C
C========================================================================
C
C---- STANDARD MOD.AMOD FUNCTIONS OF FORTRAN ARE NOT IN COMFORT
C----      BUT MAY BE ACCESSED THROUGH FUNCTIONS LINK AND ALINK
C------------------------------------------------------------
C
C     J=MOD(23,10)
C
C---- MOD RETURNS INTEGER RESULT SO USE FUNCTION LINK
      J=LINK('MOD     ',23,10)
      WRITE(6,3)J
      WRITE(3,3)J
    3 FORMAT(' MOD(23,10) IS ',I3)
C------------------------------------------------------------
```

(continued)

```
C
C      R=AMOD(12.34,1.0)
C
C---- AMOD RETURNS REAL** RESULT SO USE FUNCTION ALINK
      B=ALINK('AMOD    ',12,34.1.0)
      WRITE(6,4)B
      WRITE(3,4)B
    4 FORMAT(' AMOD(12.34,1.0) IS ',F5.3)
C------------------------------------------------------------
      END
----COMMAND----GO    ----
TODAY IS     3 AUG 78
MOD(23,10) IS    3
AMOD(12,34,1.0) IS 0.340
----COMMAND----STOP----
```