UNIVERSITY OF EDINBURGH

Edinburgh Regional Computing Centre

# System Library Manual for Edinburgh 2900 Compilers

First Edition
December 1978

# CONTENTS

## New Routine OPENSM

A new routine OPENSM has been written, which can be used instead of SMADDR. There is no intention to phase out SMADDR. The routine must be specified:

```
        externalroutinespec OPENSM(integer CHAN, MODE, c
                                   integername AD, LENGTH)
```

CHAN is used to specify the channel number (in the range 1-80) used for the DEFINE call for the file being used.

MODE can take one of the following values:

- 0 connect file in any mode
- 1 connect file for reading only
- 2 connect file for reading and writing

AD returns the address of the start of data (exactly the same as the result of a call of SMADDR) or zero if the file cannot be connected in the requested mode.

LEN returns the length of the data in the file in bytes, if the connection was successful (as with SMADDR)

Notes

This routine will be useful in several situations e.g.,

* When a user does not want to alter the contents of a file protection is provided by specifying read mode.

* When a user wants to write to a file belonging to another user then this routine allows him to check that he has write access permission, and that no other user has the file connected in a conflicting mode.

* It allows a user to deal in a controlled way with a failure to connect a file.

# PREFACE

This manual describes the standard libraries available with the Edinburgh ALGOL, FORTRAN and IMP compilers. These compilers are available on the ERCC 2970 EMAS service and the RCO 2980 VME/B service. The ALGOL and FORTRAN compilers are also marketed by ICL for use on VME/B and VME/K operating systems as 'ALGOL(E)' and 'FORTRAN(G)' respectively.

The first three sections of this manual define in detail the procedures available in Algol, the intrinsic, mathematical and generic functions and subprograms available in Fortran and the routines, functions and maps available in Imp. The final section explains in detail mixed language programming between Algol, Fortran and Imp.

As well as detailed specifications of each of the routines, there is a quick reference table preceding each detailed set of specifications.

Further information on the use of these functions can be found in:-

| | |
|---|---|
| Edinburgh ALGOL Language Manual | (First Edition June 1976) |
| Edinburgh 2900 FORTRAN Language Manual | (First Edition 1979) |
| Edinburgh IMP Language Manual | (Second Edition June 1974) |
| ICL ALGOL(E):LANGUAGE TP6855 | (First Edition September 1976) |
| ICL FORTRAN(G):LANGUAGE TP6856 | (Second Edition April 1978) |

Acknowledgements are due to a number of Regional Computing Centre Staff who have contributed to this manual, including G. E. Millard, Miss Dorothy Kidd, J.M. Murison, and N. Hamilton-Smith.

Lynda Griffiths
December 1978

# 1

# ALGOL

## 1.1 REFERENCE TABLE AND INDEX FOR ALGOL LIBRARY

| Name | Type | Parameters | Page No. |
|---|---|---|---|
| ABS(E) | rp | value E; real E; | 1-3 |
| ARCTAN(E) | rp | value E; real E; | 1-3 |
| CLOSEDA(CH) | p | value CH; integer CH; | 1-3 |
| CLOSESQ(CH) | p | value CH; integer CH; | 1-4 |
| CLOSE STREAM(CH) | p | value CH; integer CH; | 1-4 |
| CODE(X) | ip | string X; | 1-4 |
| COPYTEXT(ST) | p | string ST; | 1-5 |
| COS(E) | rp | value E; real E; | 1-5 |
| CPUTIME | rp | None | 1-5 |
| ENTIER(E) | ip | value E; real E; | 1-6 |
| EPSILON | rp | None | 1-6 |
| EXP(X) | rp | value X; real X; | 1-6 |
| FAULT(ST,R) | p | value R; string ST; real R; | 1-7 |
| GETDA(CH,SECT,A) | p | value CH; integer CH,SECT; real/integer/Boolean arrayname A; | 1-7 |
| GETSQ(CH,A) | p | value CH; integer CH; real/integer/Boolean arrayname A; | 1-8 |
| IABS(E) | ip | value E; integer E; | 1-8 |
| INCHAR(CH,ST,INT) | p | value CH; integer CH,INT; string ST; | 1-9 |
| ININTEGER(CH,I) | p | value CH; integer CH,I; | 1-10 |
| INREAL(CH,X) | p | value CH; integer CH; real X; | 1-11 |
| LENGTH(ST) | ip | string ST; | 1-11 |
| LN(E) | rp | value E; real E; | 1-12 |
| MAXINT | ip | None | 1-12 |
| MAXREAL | rp | None | 1-12 |
| MINREAL | rp | None | 1-12 |
| MONITOR | p | None | 1-13 |
| NEWLINE | p | None | 1-13 |
| NEWLINES(N) | p | value N; integer N; | 1-13 |
| NEWPAGE | p | None | 1-13 |
| NEXTCH | ip | None | 1-14 |
| NEXTSYMBOL | ip | None | 1-14 |
| OPENDA(CH) | p | value CH; integer CH; | 1-15 |
| OPENSQ(CH) | p | value CH; integer CH; | 1-15 |
| OUTCHAR(CH,ST,INT) | p | value CH,INT; integer CH,INT; string ST; | 1-16 |

| Name | Type | Parameters | Page No. |
|---|---|---|---|
| OUTINTEGER(CH,I) | p | value CH,I; integer CH,I; | 1-16 |
| OUTPUT(QU) | p | value QU; real QU; | 1-17 |
| OUTREAL(CH,X) | p | value CH,X; integer CH; real X; | 1-17 |
| OUTSTRING(CH,ST) | p | value CH; integer CH; string ST; | 1-17 |
| OUTTERMINATOR(CH) | p | value CH; integer CH; | 1-18 |
| PAPERTHROW | p | None | 1-18 |
| PRINT(Q,M,N) | p | value Q,M,N; real/integer Q; integer M,N; | 1-19 |
| PRINTCH(CH) | p | value CH; integer CH; | 1-20 |
| PRINTSTRING(ST) | p | string ST; | 1-20 |
| PRINTSYMBOL(I) | p | value I; integer I; | 1-20 |
| PRINT1900(QU,M,N) | p | value QU,M,N; real/integer QU; integer M,N; | 1-21 |
| PUTDA(CH,SECT,A) | p | value CH; integer CH,SECT; real/integer/Boolean arrayname A; | 1-22 |
| PUTSQ(CH,A) | p | value CH; integer CH; real/integer/Boolean arrayname A; | 1-23 |
| READ | rp | None | 1-23 |
| READBOOLEAN | Bp | None | 1-24 |
| READCH | ip | None | 1-24 |
| READSYMBOL(I) | p | integer I; | 1-25 |
| READ1900 | rp | None | 1-25 |
| RWNDSQ(CH) | p | value CH; integer CH; | 1-26 |
| SELECT INPUT(CH) | p | value CH; integer CH; | 1-26 |
| SELECT OUTPUT(CH) | p | value CH; integer CH; | 1-26 |
| SIGN(E) | ip | value E; real E; | 1-27 |
| SIN(E) | rp | value E; real E; | 1-27 |
| SKIPCH | p | None | 1-27 |
| SPACE | p | None | 1-28 |
| SPACES(N) | p | value N; integer N; | 1-28 |
| SQRT(E) | rp | value E; real E; | 1-28 |
| STOP | p | None | 1-28 |
| WRITE BOOLEAN(QU) | p | value QU; Boolean QU; | 1-29 |
| WRITE TEXT(ST) | p | string ST; | 1-29 |

| | | | |
|---|---|---|---|
| where | B | means | Boolean |
| | i | | integer |
| | p | | procedure |
| | r | | real |

## ABS

DEFINITION: This real procedure gives the absolute value, or modulus, of the real quantity specified on entry.

CALL: ABS (E)

E is a real parameter called by value.

ERROR CONDITIONS: None.

## ARCTAN

DEFINITION: This real procedure gives the principal value of the angle whose tangent is specified on entry.

CALL: ARCTAN (E)

E is a real parameter called by value.

The value in radians of the angle whose tangent is specified by E is returned.

ERROR CONDITIONS: None.

## CLOSEDA

DEFINITION: This procedure closes the direct access file on the channel specified on entry.

CALL: CLOSEDA(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

As all DAFILES are closed automatically at the end of a program this procedure is rarely required; but if the file is closed during the run it may not be used again without reopening.

ERROR CONDITIONS: If a file is closed twice, or an attempt is made to close a file which has been neither opened nor defined, the program terminates with an appropriate message.

## CLOSESQ

DEFINITION: This procedure closes the sequential file on the channel specified on entry.

CALL: CLOSESQ(CH)

CH is an integer parameter called by value which specifies the channel number, and whose value must be in the range allowed by the operating system being used.

As all sequential files are closed automatically at the end of a program this procedure is rarely required; but if the file is closed during the run, it may not be used again without reopening.

ERROR CONDITIONS: If a file is closed twice, an attempt is made to close a file which has not been opened, or an attempt is made to close a file which has been neither opened nor defined, the program terminates with an appropriate message.

## CLOSE STREAM

DEFINITION: This procedure closes the user-defined stream on the channel specified on entry.

CALL: CLOSE STREAM (CH)

CH is an integer parameter called by value giving the number of the stream to be closed.

The stream CH is closed such that on the next call of either SELECT INPUT or SELECT OUTPUT for that stream, the stream is reset to the start. Note that CH must not be the current input or output stream.

ERROR CONDITIONS: If the stream is not defined on entry to the procedure, if an attempt is made to close a stream not in the range allowed by the operating system being used, or if CH is a current stream, then the program terminates with the following message:

```
STREAM NOT DEFINED
```

## CODE

DEFINITION: This integer procedure provides the code value of the character X which must be a single character and must be enclosed in string quotes.

CALL: CODE (X)

X is a single character string parameter.

This procedure enables the programmer to manipulate the character information without needing to know the internal codes employed for characters, and greatly facilitates the transfer of ALGOL programs between computers using different internal codes. As in PRINTSTRING, _ and \ will give the internal code for space and newline respectively. The two character string EL will also give the newline code.

ERROR CONDITIONS: None.

## COPYTEXT

DEFINITION: This procedure transfers symbols from the input stream to the output stream up to but not including the string parameter specified on entry.

CALL: COPYTEXT (ST)

ST is a string parameter.

Symbols are transferred from the input stream to the output stream until the string ST is encountered. The string itself is not output.

ERROR CONDITIONS: None.

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## COS

DEFINITION: This real procedure gives the principal value of the cosine of the quantity specified on entry. This quantity must be in radians.

CALL: COS(E)

E is a real parameter called by value.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than $3.53 \times 10^{15}$, the result would be wholly inaccurate, and the program terminates with the following message:

```
COS ARG OUT OF RANGE
```

## CPUTIME

DEFINITION: This real procedure gives the total CPU time used from an arbitrary starting point.

CALL: CPUTIME

No parameters.

The total CPU time used since an arbitrary starting point is returned as a positive real number having seconds as its unit of time. Note that there is a CPU time overhead in calling the function itself.

ERROR CONDITIONS: None.

## ENTIER

DEFINITION: This integer procedure gives the largest integer not greater than the value of the quantity specified on entry.

CALL: ENTIER(E)

E is a real parameter called by value.

The value of the largest integer not greater than the value of E is returned.

```
e.g.  ENTIER(2.7)= 2
      ENTIER(-3.1)=-4
```

ERROR CONDITIONS: Integer overflow will occur if E is outside the range of valid integers.

## EPSILON

DEFINITION: This real procedure gives the smallest positive real number such that the computational result of (1.0 + EPSILON) is greater than 1.0, and the computational result of (1.0 - EPSILON) is less than 1.0.

CALL: EPSILON

No parameters.

The value of the smallest positive real number to satisfy the following conditions is returned:

```
1.0 + EPSILON > 1.0
1.0 - EPSILON < 1.0
```

ERROR CONDITIONS: None.

## EXP

DEFINITION: This real procedure gives the value of 'e' raised to the power of the quantity specified on entry.

CALL: EXP(X)

X is a real parameter called by value.

The value of 'e' raised to the power X is returned.

If the argument of this function is less than -180.218 then zero is returned.

ERROR CONDITIONS: If the argument of this function is greater than 174.673, the program terminates with the following message:

```
EXP ARG OUT OF RANGE
```

## FAULT

DEFINITION: This procedure outputs the string ST, prints the parameter R on the current output stream and then terminates program execution.

CALL: FAULT(ST,R)

| | |
|---|---|
| ST | is a string parameter. |
| R | is a real parameter called by value. |

ERROR CONDITIONS: None.

## GETDA

DEFINITION: This procedure reads data from the specified channel into the specified area from the specified block on the file.

CALL: GETDA (CH,SECT,A)

| | |
|---|---|
| CH | is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used. |
| SECT | is an integer parameter whose value on entry specifies the number of the first record to be read and, on exit, contains the number of the last record read by this call. |
| A | is the name of the array to be read and can be of any type. |

e.g. <u>real</u>/<u>integer</u>/<u>Boolean</u> <u>arrayname</u> A

Information is read from file CH starting at record SECT and is written into ARRAY A.

ERROR CONDITIONS: If an attempt is made to access a record on the file which does not exist, the program terminates with the message:

RECORD NUMBER OUT OF RANGE

If the specified channel is not open, the program terminates with:

FILE NOT OPEN

If an invalid record size has been specified in a file definition, the program terminates with the message:

INVALID RECORD SIZE

If an input or output request is not consistent with the file definition, the program terminates with the message:

INVALID OPERATION ON FILE

If an attempt is made to read from a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

On creation the complete file is filled with the unassigned pattern.

## GETSQ

DEFINITION: This procedure reads one record from the channel specified on entry into the array specified on entry.

CALL: GETSQ(CH,A)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

A is the name of the array into which the record is read and can be of any type:

real/integer/Boolean arrayname A

ERROR CONDITIONS: If the record read is larger than the array then the record will be truncated. If the record is smaller, then the content of the remainder of the array is undefined.

If an attempt is made to read a record before opening channel CH then the program terminates with the message:

```
FILE NOT OPEN
```

If an attempt is made to read beyond the end of the file then the program terminates with the message:

```
INPUT ENDED
```

If an attempt is made to read from a file with an invalid channel number or one which has not been defined the program terminates with an appropriate error message.

## IABS

DEFINITION: This integer procedure gives the value of the modulus of the integer quantity specified on entry.

CALL: IABS(E)

E is an integer parameter called by value.

ERROR CONDITIONS: Integer overflow occurs if an attempt is made to take the modulus of the largest negative integer.

## INCHAR

DEFINITION: This procedure uses the SELECT INPUT procedure to select the input stream CH, unless it is already selected, and then sets INT to the value corresponding to the first position in string ST of the current input character.

CALL: INCHAR(CH,ST,INT)

| | |
|---|---|
| CH | is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used. |
| ST | is a STRING parameter. |
| INT | is an integer variable which is assigned the value corresponding to the first position in ST of the current input character.<br><br>INT is set to 1 if the first character of the string corresponds to the current character on the input stream. |

ERROR CONDITIONS: If the current character is not given in the string, ST, then INT is set to zero. The stream pointer is updated to point to the next character.

If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

```
INPUT ENDED
```

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

```
SUBSTITUTE CHARACTER IN DATA
```

This fault will occur on attempting to read the first character of the line, not necessarily the incorrect character.

## ININTEGER

DEFINITION: This procedure uses the SELECT INPUT procedure to select the input stream CH, if not already selected, and then uses procedure READ, to input the next number and assign it to I.

CALL: ININTEGER (CH,I)

| | |
|---|---|
| CH | is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used. |
| I | is an integer variable and is assigned from the next number read on the currently selected input stream. |

The next number in the input stream is read and assigned to integer I.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, <u>not</u> necessarily the incorrect character.

If a non-numeric symbol is found when not expected then the program terminates with the message:

SYMBOL IN DATA

followed by the invalid symbol.

## INREAL

DEFINITION: This procedure uses the SELECT INPUT procedure to select the input stream CH, if not already selected, and then uses procedure READ to input the next number and assign it to X.

CALL: INREAL(CH,X)

| | |
|---|---|
| CH | is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used. |
| X | is a real variable and is assigned from the next number read on the currently selected input stream. |

The next number in the input stream is read and assigned to real X.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

If a non-numeric symbol is found when not expected then the program terminates with the message:

SYMBOL IN DATA

followed by the invalid symbol.

## LENGTH

DEFINITION: This integer procedure returns the number of characters in the open string, ST, excluding the enclosing string quotes.

CALL: LENGTH (ST)

| | |
|---|---|
| ST | is a string parameter. |

ERROR CONDITIONS: None

## LN

DEFINITION: This real procedure returns the value of the logarithm to the base 'e' of the quantity specified on entry.

CALL: LN(E)

E is a real parameter called by value.

ERROR CONDITIONS: If E is negative the program terminates with the following message:

```
LOG ARG NEGATIVE
```

If E is zero the program terminates with the message:

```
LOG ARG ZERO
```

## MAXINT

DEFINITION: This integer procedure gives the maximum allowable positive integer.

CALL: MAXINT

No parameters.

ERROR CONDITIONS: None.

## MAXREAL

DEFINITION: This real procedure gives the maximum allowable positive real number.

CALL: MAXREAL

No parameters.

ERROR CONDITIONS: None.

## MINREAL

DEFINITION: This real procedure gives the minimum allowable positive real number that is distinguishable from zero.

CALL: MINREAL

No parameters.

ERROR CONDITIONS: None.

## MONITOR

DEFINITION: This procedure can be called to invoke the diagnostic package without terminating execution of the program.

CALL: MONITOR

No parameters.

ERROR CONDITIONS: None.

## NEWLINE

DEFINITION: This procedure outputs one 'newline' character to the currently selected output stream.

CALL: NEWLINE

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## NEWLINES

DEFINITION: This procedure outputs N newline characters to the currently selected output stream. If N <= 0 then the procedure has no effect.

CALL: NEWLINES(N)

N is an integer parameter called by value.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## NEWPAGE

DEFINITION: This procedure outputs to the currently selected output stream a command to throw to the head of a new page.

CALL: NEWPAGE

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## NEXTCH

DEFINITION: This integer procedure reads one character from the currently selected input stream, but leaves the input stream positioned at that character so that the character may be read again with a call of READCH or a further call of NEXTCH.

CALL: NEXTCH

No parameters.

Any number of consecutive calls of NEXTCH on the same channel will all obtain the same character.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

NOTE: This is a 1900 ALGOL compatible procedure provided for the convenience of programmers with working 1900 programs.

## NEXTSYMBOL

DEFINITION: This integer procedure reads one character from the currently selected input stream but leaves the input stream positioned at that character.

CALL: NEXTSYMBOL

No parameters.

Any number of consecutive calls of NEXTSYMBOL on the same channel will all obtain the same character.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

## OPENDA

DEFINITION: This procedure opens the direct access file on channel number CH, where CH must be in the range allowed by the operating system being used.

CALL: OPENDA(CH)

CH is an integer parameter called by value, which specifies the channel number to be opened and whose value must be in the range allowed by the operating system being used.

The file is opened for reading or writing.

ERROR CONDITIONS: If a file is opened twice without being closed, the program terminates with the following message:

```
FILE ALREADY OPEN
```

If an attempt is made to open a file with an invalid channel number or a channel which has not been defined, the program terminates with an appropriate error message.

Although the channel number must lie in the range appropriate to the operating system being used it must not conflict with any channel numbers previously assigned to either stream I/O or to sequential files.

The user must declare his direct access file requirements to the operating system as specified in the appropriate User's Guide.

## OPENSQ

DEFINITION: This procedure opens a binary sequential I/O channel. It must be called before GETSQ or PUTSQ is called for that channel.

CALL: OPENSQ(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

The file is made ready for reading or writing, but note that both options may not be exercised. If a file is closed in a program and then reopened, it will be reset to the start.

ERROR CONDITIONS: If a file is opened twice without being closed, the program is terminated with the message:

```
FILE ALREADY OPEN
```

If an attempt is made to open a file with an invalid channel number or a channel which has not been defined, the program terminates with an appropriate error message.

Although the logical channel number must lie in the range appropriate to the system being used it must not conflict with any channel numbers assigned to either stream I/O or direct access files.

The user must declare his sequential file requirements to the operating system as specified in the appropriate User's Guide.

## OUTCHAR

DEFINITION: This procedure uses SELECT OUTPUT to select the output stream, CH, if not already selected, and then outputs the character in the string, ST, to which the value of INT points.

CALL: OUTCHAR(CH,ST,INT)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

ST is a string parameter, one character of which is to be output.

INT is an integer parameter called by value which points to a character within the string ST.

If INT has a value 1, the first character of the string, ST, is output.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## OUTINTEGER

DEFINITION: This procedure uses the SELECT OUTPUT procedure to select the output stream, CH, if not already selected, and outputs I in the form PRINT(I,10,0) followed by a semi-colon and a newline.

CALL: OUTINTEGER(CH,I)

CH is an integer parameter called by value which specifies the channel number, and whose value must be in the range allowed by the operating system being used.

I is an integer parameter called by value which is to be output to the currently selected output stream.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## OUTPUT

DEFINITION: This procedure outputs the value of QU in floating point form D&E where the mantissa D is written with 10 significant figures and the exponent E consists of two digits preceded by a negative sign or a space representing + and followed by a semi-colon and a newline.

CALL: OUTPUT(QU)

QU is a real parameter called by value which is to be output in floating point form.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## OUTREAL

DEFINITION: This procedure uses the SELECT OUTPUT procedure to select the output stream, CH, if not already selected, and outputs X in the form PRINT(X,0,10) followed by a semi-colon and a newline.

CALL: OUTREAL(CH,X)

CH is an integer parameter called by value which specifies the channel number, and whose value must be in the range allowed by the operating system being used.

X is a real parameter called by value which is to be output to the currently selected output stream.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

## OUTSTRING

DEFINITION: This procedure uses SELECT OUTPUT to select the output stream, CH, if not already selected, and then outputs the string, ST, followed by a semi-colon and a newline.

CALL: OUTSTRING(CH,ST)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

ST is a string parameter to be output.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

## OUTTERMINATOR

DEFINITION: This procedure uses SELECT OUTPUT to select the output stream, CH, if not already selected, and then outputs a semi-colon followed by a newline.

CALL: OUTTERMINATOR(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## PAPERTHROW

DEFINITION: This procedure outputs to the currently selected output stream a command to throw to the head of a new page.

CALL: PAPERTHROW

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## PRINT

DEFINITION: This procedure outputs to the currently selected output stream the value of Q, preceded by a sign, in the format determined by the parameters M and N.

CALL: PRINT(Q,M,N)

Q is a real or integer parameter called by value which specifies the number to be output.

M and N are integer parameters called by value which determine the format of the number as follows:

M=0,N#0 The number is output in the floating point form D&E, where the mantissa D is written to N+1 significant digits such that 1 <= D < 10, and the exponent E consists of two digits with a negative sign or a space representing +. The number always occupies N+7 character positions in all.

```
e.g.    M=0 and N=5:
        -1.23456& 10
         3.45678&-12
         1.00000&  2
```

M#0,N#0 The number is output in fixed-point form with M digits to the left of the decimal point and N digits to the right. The number occupies M+N+2 character positions in all.

```
e.g.    M=3 and N=2:
         123.45
          22.25
          -1.00
```

M#0,N=0 The number is output in integer form occupying M+1 character positions in all.

```
e.g.    M=4 and N=0:
        55555
        -1234
           10
```

In each format, the total count of character positions includes a sign character which is a minus or a space.

If M#0 and the number has more than M digits in the integral part, further character positions will be used to accommodate the number. This may cause following numbers to be displaced to the right when the results are printed.

If M#0 and the number has less than M digits in the integral part, spaces are inserted before the sign character to fill up to the required number of character positions.

For numbers whose modulus is less than 1, the zero before the decimal point is printed; that is, it is not replaced by a space, for example, 0.001. The fractional part of a number is rounded, in the decimal form, to the appropriate number of decimal places.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## PRINTCH

DEFINITION: This procedure outputs one character (defined by the n least significant bits of CH) to the currently selected output stream: n is seven under VME/B and eight on EMAS.

CALL: PRINTCH(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## PRINTSTRING

DEFINITION: This procedure outputs the given string to the currently selected output stream.

CALL: PRINTSTRING(ST)

ST is a string parameter.

Since spaces and newlines are ignored in ALGOL, it is necessary to use special characters within the string to represent space and newline. The space is represented by _ and newline by \. The string is stored with _ and \ represented by their ISO internal code and the substitution is done by the PRINTSTRING procedure when the string is output.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## PRINTSYMBOL

DEFINITION: This procedure outputs one character defined by the n least significant bits of I to the currently selected output stream: n is seven under VME/B and eight on EMAS.

CALL: PRINTSYMBOL(I)

I is an integer variable called by value which specifies the character to be output.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## PRINT1900

DEFINITION: This procedure outputs to the currently selected output stream the value of QU followed by two spaces.

CALL: PRINT1900(QU,M,N)

QU is a real or integer parameter called by value which specifies the number to be output.

M and N are integer parameters called by value which determine the format of the number as follows:

M=0,N#0 The number is output in the floating point form D&E, where the mantissa D is written to N+1 significant digits such that 1 <= D < 10, and the exponent E consists of two digits with a negative sign or a space representing +. The number always occupies N+9 character positions in all.

```
e.g.     M=0 and N=5:
         -1.23456& 10
          3.45678&-12
          1.00000&  2
```

M#0,N#0 The number is output in fixed-point form with M digits to the left of the decimal point and N digits to the right. The number occupies M+N+4 character positions in all.

```
e.g.     M=3 and N=2:
          123.45
           22.25
           -1.00
```

M#0,N=0 The number is output in integer form occupying M+3 character positions in all.

```
e.g.     M=4 and N=0:
         55555
         -1234
            10
```

In each format, the total count of character positions includes a sign character which is a minus or a space, and two spaces after the last digit.

If M#0 and the number has more than M digits in the integral part, further character positions will be used to accommodate the number. This may cause following numbers to be displaced to the right when the results are printed.

If M#0 and the number has less than M digits in the integral part, spaces are inserted before the sign character to fill up to the required number of character positions.

For numbers whose modulus is less than 1, the zero before the decimal point is printed; that is, it is not replaced by a space, for example, 0.001 . The fractional part of a number is rounded, in the decimal form, to the appropriate number of decimal places.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## PUTDA

DEFINITION: This procedure is used to write data from one array in store to a file.

CALL: PUTDA(CH,SECT,A)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

SECT is the name of an integer variable which on entry should contain the number of the first record to be written.
On exit it will contain the number of the record after the last record written by this call.

A is the name of the array to be written and can be of type <u>real</u>, <u>integer</u> or <u>boolean</u>.

A call of PUTDA will result in one or more records being written depending on the number of bytes in the array defined. If the array written is not an exact multiple of the defined record size then the contents of the end of the last record written will be undefined.

ERROR CONDITIONS: If an attempt is made to access a non-existent record on the specified file, the program terminates with the message:

RECORD NUMBER OUT OF RANGE

If the specified channel is not open, the program terminates with:

FILE NOT OPEN

If an attempt is made to write to a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

If an invalid record size has been specified in a file definition, the program terminates with the message:

INVALID RECORD SIZE

If an input or output request is not consistent with the file definition, the program terminates with the message:

INVALID OPERATION ON FILE

## PUTSQ

DEFINITION: This procedure outputs one logical record, for each call on PUTSQ, the length of which is determined by the length of the array to be output.

CALL: PUTSQ(CH,A)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

A is the name of the array in which the data will be found and can be of type real, integer or Boolean

ERROR CONDITIONS: If a minimum and/or maximum record size for the sequential file has been defined and the length of the array to be written is incompatible with this record size then the call on PUTSQ will be faulted.

If the specified channel is not open then the program terminates with the following message:

FILE NOT OPEN

## READ

DEFINITION: This real procedure reads numerical data from the currently selected input stream, ignoring multiple space, newline or newpage characters before the start of the number, and terminates on reading the first character which is neither a digit nor a correctly placed exponent or sign. The symbols & and @ are used as representation of $_{10}$

CALL: READ

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

If a non-numeric symbol is found when not expected then the program terminates with the message:

SYMBOL IN DATA

followed by the invalid symbol.

## READBOOLEAN

DEFINITION: This Boolean procedure searches the current input stream for one of the basic symbols true or false.

CALL: READ BOOLEAN

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## READCH

DEFINITION: This integer procedure reads one character from the currently selected input stream. The procedure reads any characters from the input, including the 'invisible' characters which READ SYMBOL ignores.

CALL: READCH

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## READSYMBOL

DEFINITION: This procedure reads one symbol from the currently selected input stream and assigns it to the parameter I, which must be a variable. Only visible symbols are read by this procedure.

CALL: READSYMBOL(I)

I is an integer variable which is assigned when a symbol is read from the currently selected input stream.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

## READ1900

DEFINITION: This real procedure is a version of the READ procedure which allows a single space within the number. It also ignores non-numeric characters before the start of the number and accepts '10' and E in place of & or @ to introduce an exponent.

CALL: READ1900

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

INPUT ENDED

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

SUBSTITUTE CHARACTER IN DATA

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

If a non-numeric symbol is found when not expected then the program terminates with the message:

SYMBOL IN DATA

followed by the invalid symbol.

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## RWNDSQ

DEFINITION: This procedure resets the sequential file specified by the value of CH to the start of the first record.

CALL: RWNDSQ(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

ERROR CONDITIONS: None.

## SELECT INPUT

DEFINITION: This procedure selects input stream CH: subsequent calls of input procedures take information from stream CH unless and until a further call of this procedure selects a different stream.

CALL: SELECT INPUT(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

Only one input stream may be selected at any time, and it may not be the same as the output stream. If a call of SELECT INPUT is made part way through reading a line of input the remainder of that line is lost.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

```
INPUT ENDED
```

## SELECT OUTPUT

DEFINITION: This procedure selects output stream CH; subsequent calls of output procedures send information to stream CH unless and until a further call of this procedure selects a different stream.

CALL: SELECT OUTPUT(CH)

CH is an integer parameter called by value which specifies the channel number and whose value must be in the range allowed by the operating system being used.

Only one output stream may be selected at any one time, and it may not be the same as the input stream. If a call of SELECT OUTPUT is made part way through a line of output, that partial line will be output as though the call of NEWLINE immediately preceded the call of SELECT OUTPUT.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## SIGN

DEFINITION: This integer procedure gives the sign of the value of E as an integer:

```
SIGN(E) = +1 if E > 0
SIGN(E) = -1 if E < 0
SIGN(E) =  0 if E = 0
```

CALL: SIGN(E)

E is a real parameter called by value.

ERROR CONDITIONS: None.

## SIN

DEFINITION: This real procedure gives the sine of E, where E is expressed in radians.

CALL: SIN(E)

E is a real parameter called by value.

ERROR CONDITIONS: If the modulus of this argument is greater than $3.53 \times 10^{15}$, the result would be wholly inaccurate and the program terminates with the following message:

```
SIN ARG OUT OF RANGE
```

## SKIPCH

DEFINITION: This procedure skips over one character of the input stream without reading it.

CALL: SKIPCH

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the following message:

```
INPUT ENDED
```

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this procedure the program terminates with the following message:

```
SUBSTITUTE CHARACTER IN DATA
```

This fault will occur on attempting to read the first character of the line, NOT necessarily the incorrect character.

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## SPACE

DEFINITION: This procedure outputs to the currently selected output stream one horizontal space.

CALL: SPACE

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## SPACES

DEFINITION: This procedure outputs to the currently selected output stream N horizontal spaces, each space being equivalent to one character.

CALL: SPACES(N)

N is an integer parameter called by value.

If N <= 0 then the procedure has no effect.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

```
OUTPUT EXCEEDED
```

## SQRT

DEFINITION: This real procedure gives the positive square root of E for E > 0.

CALL: SQRT(E)

E is a real parameter called by value.

ERROR CONDITIONS: If a negative argument is passed to this routine, the program terminates with the following message:

```
SQRT ARG NEGATIVE
```

## STOP

DEFINITION: This procedure causes execution to cease by arranging a branch to the <u>end</u> corresponding to the first <u>begin</u> of the program.

CALL: STOP

No parameters.

ERROR CONDITIONS: None.

## WRITE BOOLEAN

DEFINITION: This procedure outputs the value of the Boolean expression to the currently selected output stream as <u>true</u> followed by two spaces, or <u>false</u> followed by one space.

CALL: WRITE BOOLEAN(QU)

QU is a Boolean parameter called by value.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

## WRITE TEXT

DEFINITION: This procedure outputs the string ST to the currently selected output stream.

CALL: WRITE TEXT(ST)

ST is a string parameter.

The editing characters S for space, C for newline, and P for newpage are accepted, enclosed in further string quotes. An editing character may be preceded by an unsigned integer N to indicate N such characters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the following message:

OUTPUT EXCEEDED

NOTE: This is a 1900 ALGOL-compatible procedure provided for the convenience of programmers with working 1900 programs.

# 2

# FORTRAN

## 2.1 Introduction

The Fortran compiler on ICL 2900 series supports generic functions, which provide an automatic function selection facility. This facility allows the programmer to use a single generic name when requesting a Fortran-supplied function which has several names, depending on argument type. The proper function is selected by the Fortran Compiler, based on the type and length of the arguments of the function. With this facility the programmer can, for example, use the generic name SIN to refer to any sine routine, rather than explicitly calling SIN for REAL*4 arguments, DSIN for REAL*8 arguments, CSIN for COMPLEX*8 arguments etc.

If the specific name is used, then the arguments must be of the correct type and size otherwise a compile time fault will be reported.

Specific function names, rather than generic names, must be passed as arguments to external procedures. The automatic function selection facility will not substitute the appropriate function name for the generic name in an argument list. (There is no way to make such a selection, since the name being passed as an argument has no arguments of its own).

In the table which follows where no specific name appears, then the generic name is the only one which can be used.

The additional supplied subroutines, tabulated in section 2.2.3, are available on the ERCC 2970 EMAS service and the RCO 2980 VME/B service, but are not necessarily available on other installations who have received the FORTRAN(G) compiler from ICL.

## 2.2 REFERENCE TABLE AND INDEX FOR FORTRAN LIBRARY

### 2.2.1 GENERIC FUNCTIONS

| Generic Name | Function | Specific Name | No. of Arg. | Type of Arguments | Type of Function Value | Page |
|---|---|---|---|---|---|---|
| ABS | Absolute value | IABS<br>ABS<br>DABS<br>QABS<br>CABS<br>CDABS<br>CQABS | 1<br>1<br>1<br>1<br>1<br>1<br>1 | Integer*4<br>Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Integer*4<br>Real*4<br>Real*8<br>Real*16<br>Real*4<br>Real*8<br>Real*16 | 2-8 |
| ACOS | Arccosine | ARCOS<br>DARCOS<br>QARCOS | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-8 |
| AINT | Truncation | AINT<br>DINT<br>QINT | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-8 |
| ANINT | Nearest whole number | ANINT<br>DNINT | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-9 |
| ASIN | Arcsine | ARSIN<br>DARSIN<br>QARSIN | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-9 |
| ATAN | Arctangent | ATAN<br>DATAN<br>QATAN | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-9 |
| ATAN2 | Arctangent | ATAN2<br>DATAN2<br>QATAN2 | 2<br>2<br>2 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-10 |
| CMPLX | Express two real args in complex form | CMPLX<br>DCMPLX<br>QCMPLX | 2<br>2<br>2 | Real*4<br>Real*8<br>Real*16 | Complex*8<br>Complex*16<br>Complex*32 | 2-10 |
| CONJG | Obtain conjugate of complex arg. | CONJG<br>DCONJG<br>QCONJG | 1<br>1<br>1 | Complex*8<br>Complex*16<br>Complex*32 | Complex*8<br>Complex*16<br>Complex*32 | 2-10 |
| COS | Trigonometric cosine (argument in radians) | COS<br>DCOS<br>QCOS<br>CCOS<br>CDCOS<br>CQCOS | 1<br>1<br>1<br>1<br>1<br>1 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | 2-11 |
| COSH | Hyperbolic cosine | COSH<br>DCOSH<br>QCOSH | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-11 |

| Generic Name | Function | Specific Name | No. of Arg. | Type of Arguments | Type of Function Value | Page |
|---|---|---|---|---|---|---|
| COTAN | Trigonometric cotangent (argument in radians) | COTAN<br>DCOTAN<br>QCOTAN | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-11 |
| DBLE | Convert to double precision | DFLOAT<br>DBLE<br><br>DBLEQ | 1<br>1<br>1<br>1 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Real*8<br>Real*8<br>Real*8<br>Real*8 | 2-12 |
| DIM | Positive difference | IDIM<br>DIM<br>DDIM<br>QDIM | 2<br>2<br>2<br>2 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | 2-12 |
| ERF | Error function | ERF<br>DERF<br>QERF | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-13 |
| ERFC | Complemented error function | ERFC<br>DERFC<br>QERFC | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-13 |
| EXP | Exponential | EXP<br>DEXP<br>QEXP<br>CEXP<br>CDEXP<br>CQEXP | 1<br>1<br>1<br>1<br>1<br>1 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | 2-13 |
| GAMMA | Gamma function | GAMMA<br>DGAMMA | 1<br>1 | Real*4<br>Real*8 | Real*4<br>Real*8 | 2-14 |
| IMAG | Obtain imaginary part of complex argument | AIMAG<br>DIMAG<br>QIMAG | 1<br>1<br>1 | Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16 | 2-16 |
| INT | Conversion to integer | <br>IFIX<br>IDINT<br>IQINT | 1<br>1<br>1<br>1 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Integer*4<br>Integer*4<br>Integer*4 | 2-16 |
| LGAMMA | Log-gamma | ALGAMA<br>DLGAMA | 1<br>1 | Real*4<br>Real*8 | Real*4<br>Real*8 | 2-17 |
| LOG | Natural logarithm | ALOG<br>DLOG<br>QLOG<br>CLOG<br>CDLOG<br>CQLOG | 1<br>1<br>1<br>1<br>1<br>1 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | 2-17 |

| Generic Name | Function | Specific Name | No. of Arg. | Type of Arguments | Type of Function Value | Page |
|---|---|---|---|---|---|---|
| LOG10 | Common Logarithm | ALOG10<br>DLOG10<br>QLOG10 | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-18 |
| MAX | Largest value | MAX0<br>AMAX1<br>DMAX1<br>QMAX1 | >=2<br>>=2<br>>=2<br>>=2 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | 2-18 |
| MIN | Smallest value | MIN0<br>AMIN1<br>DMIN1<br>QMIN1 | >=2<br>>=2<br>>=2<br>>=2 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | 2-18 |
| MOD | Modular arithmetic | <br>AMOD<br>DMOD<br>QMOD | 2<br>2<br>2<br>2 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | 2-19 |
| NINT | Nearest integer | NINT<br>IDNINT | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Integer*4<br>Integer*4<br>Integer*4 | 2-19 |
| QEXT | Extend precision to quad | QEXT<br>QEXTD | 1<br>1 | Real*4<br>Real*8 | Real*16<br>Real*16 | 2-19 |
| REAL | Conversion to real | FLOAT<br><br>SNGL<br><br>DREAL<br>QREAL | 1<br>1<br>1<br>1<br>1<br>1 | Integer*4<br>Real*4<br>Real*8<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*4<br>Real*4<br>Real*4<br>Real*8<br>Real*16 | 2-20 |
| SIGN | Transfer of sign | ISIGN<br>SIGN<br>DSIGN<br>QSIGN | 2<br>2<br>2<br>2 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | Integer*4<br>Real*4<br>Real*8<br>Real*16 | 2-20 |
| SIN | Trigonometric sine (argument in radians) | SIN<br>DSIN<br>QSIN<br>CSIN<br>CDSIN<br>CQSIN | 1<br>1<br>1<br>1<br>1<br>1 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | 2-20 |
| SINH | Hyperbolic sine | SINH<br>DSINH<br>QSINH | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-20 |
| SQRT | Square root | SQRT<br>DSQRT<br>QSQRT<br>CSQRT<br>CDSQRT<br>CQSQRT | 1<br>1<br>1<br>1<br>1<br>1 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | Real*4<br>Real*8<br>Real*16<br>Complex*8<br>Complex*16<br>Complex*32 | 2-21 |

| Generic Name | Function | Specific Name | No. of Arg. | Type of Arguments | Type of Function Value | Page |
|---|---|---|---|---|---|---|
| TAN | Trigonometric tangent (argument in radians) | TAN<br>DTAN<br>QTAN | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-21 |
| TANH | Hyperbolic tangent | TANH<br>DTANH<br>QTANH | 1<br>1<br>1 | Real*4<br>Real*8<br>Real*16 | Real*4<br>Real*8<br>Real*16 | 2-21 |

## 2.2.2 SPECIFIC FUNCTIONS

The following functions have no generic names and must be called by their specific name.

| Specific Name | Function | No. of Arg | Type of Arguments | Type of Function | Page |
|---|---|---|---|---|---|
| AMAX0 | Largest value | >=2 | Integer*4 | Real*4 | 2-8 |
| AMIN0 | Smallest value | >=2 | Integer*4 | Real*4 | 2-9 |
| CHAR | Conversion to type character | 1 | Integer*4 | Character | 2-10 |
| DPROD | Double precision product | 2 | Real*4 | Real*8 | 2-12 |
| HFIX | Conversion to half word | 1 | Real*4 | Integer*2 | 2-14 |
| ICHAR | Convert a single character to an integer value | 1 | Character | Integer*4 | 2-14 |
| INDEX | Location of substring $a_2$ in string $a_1$ | 2 | Character | Integer*4 | 2-16 |
| LEN | Length of character entity | 1 | Character | Integer*4 | 2-17 |
| MAX1 | Largest value | >=2 | Real*4 | Integer*4 | 2-18 |
| MIN1 | Smallest value | >=2 | Real*4 | Integer*4 | 2-19 |

## 2.2.3 ADDITIONAL SUPPLIED SUBROUTINES

A number of additional subroutines are available to users of Edinburgh FORTRAN. These are not standard, but provide useful facilities which are often available with minor differences in other implementations of FORTRAN. No special library has to be declared to access any of these routines.

| Entry | Definition | No. of Arguments | Type of Argument | Page No. |
|---|---|---|---|---|
| CPUTIME * | The total CPU time in seconds used from an arbitrary starting point | 1 | Real*8 | 2-11 |
| CTIME | Clock time as an ISO character string hh:mm:ss | 1 | Real*8 | 2-12 |
| HDATE | Date as an ISO character string dd/mm/yy | 1 | Real*8 | 2-14 |
| ICL9CECPUTIME | The total CPU time used from an arbitrary starting point | 1 | Real*8 | 2-15 |
| ICL9CEDATE | Date as an EBCDIC character string dd/mm/yy | 1 | Real*8 | 2-15 |
| ICL9CEDIAG | Routine traceback printed if DIAG option is selected at compile time | 0 |  | 2-15 |
| ICL9CEFTRACE | This subroutine may be called to switch on and off the listing of labels, subroutine entries and exits during execution of a program depending on the value of the parameter | 1 | Integer*4 | 2-15 |
| ICL9CELABELS | The 32 most recent statement labels, subprogram entries and returns encountered are printed if the LABELS option is selected at compile time | 0 |  | 2-15 |
| ICL9CETIME | Clock time as an EBCDIC character string hh:mm:ss | 1 | Real*8 | 2-16 |

* this is a real *8 function with no arguments.

## ABS

DEFINITION: This set of functions gives the value of the modulus (i.e. the absolute value) of the quantity specified on entry. For complex arguments [X=x+iy] the absolute value is SQRT($x^2+y^2$)

SPECIFIC NAMES: IABS, DABS, QABS, CABS, CDABS, CQABS.

ERROR CONDITIONS: None.

## ACOS

DEFINITION: This set of functions finds the value of the angle whose cosine is specified on entry. The value is given in radians and lies in the range 0 to + $\pi$

SPECIFIC NAMES: ARCOS, DARCOS, QARCOS.

ERROR CONDITIONS: If X is not in the range -1 <= X <= 1, the program terminates with the following message:

```
ACOS ARG OUT OF RANGE
```

## AINT

DEFINITION: This set of functions returns the value of the largest whole number less than or equal to the absolute value of the quantity specified on entry. The function takes the same sign as the argument.

SPECIFIC NAMES: DINT, QINT.

ERROR CONDITIONS: None.

## AMAX0

DEFINITION: This function gives the real value of the largest integer argument specified on entry in the parameter list.

SPECIFIC NAME: AMAX0.

ERROR CONDITIONS: None.

NOTE: AMAX0 is a specific name; there is no generic name.

## AMINO

DEFINITION: This function gives the real value of the smallest integer argument specified on entry in the parameter list.

SPECIFIC NAME: AMINO.

ERROR CONDITIONS: None.

NOTE: AMINO is a specific name; there is no generic name.

## ANINT

DEFINITION: This set of functions returns the nearest whole number to the quantity specified on entry.

```
i.e.  INT(X+0.5) if X ≥ 0
      INT(X-0.5) if X < 0
```

SPECIFIC NAME: DNINT.

ERROR CONDITIONS: None.

## ASIN

DEFINITION: This set of functions returns the value of the angle whose sine is specified on entry. The value is given in radians and lies in the range $-\frac{\pi}{2}$ to $\frac{\pi}{2}$.

SPECIFIC NAMES: ARSIN, DARSIN, QARSIN.

ERROR CONDITIONS: If X is not in the range -1 <= X <= 1, the program terminates with the following message:

```
ASIN ARG OUT OF RANGE
```

## ATAN

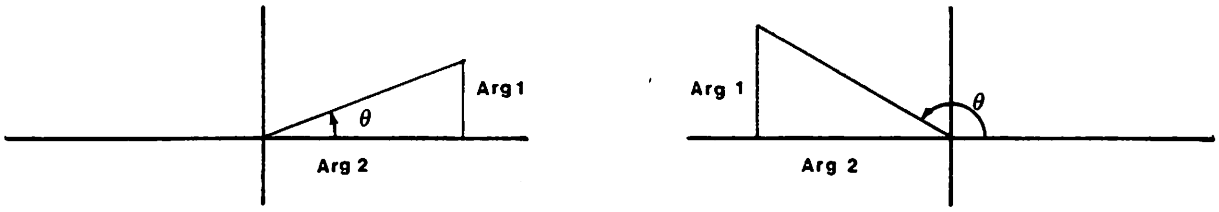
DEFINITION: This set of functions give the value of the angle whose tangent is specified on entry. The value is given in radians and lies in the range $-\pi$ to $\pi$.

SPECIFIC NAMES: DATAN, QATAN.

ERROR CONDITIONS: None.

## ATAN2

DEFINITION: This set of functions gives the value of the angle whose tangent is specified on entry.
The value in radians of the angle whose tangent is specified by 'Argl/Arg2' is returned [i.e. tan (?)=Argl/Arg2]. This value is given in radians and lies in the range $-\pi$ to $\pi$ .



SPECIFIC NAMES: DATAN2, QATAN2.

ERROR CONDITIONS: If the parameters passed to this routine are both zero, the program terminates with the following message:

```
ATAN2 ARGS ZERO
```

## CHAR

DEFINITION: This function converts the integer argument to character type: ISO or EBCDIC as appropriate.

SPECIFIC NAME: CHAR.

ERROR CONDITIONS: None.

NOTE: CHAR is a specific name; there is no generic name.

## CMPLX

DEFINITION: This set of functions expresses in complex form the two real quantities specified on entry.

SPECIFIC NAMES: DCMPLX, QCMPLX.

ERROR CONDITIONS: None.

## CONJG

DEFINITION: This set of functions obtains the conjugate of the complex quantity specified on entry. (i.e. Arg = x + i y; Conjugate = x - i y).

SPECIFIC NAMES: DCONJG, QCONJG.

ERROR CONDITIONS: None.

## COS

DEFINITION: This set of functions gives the value of the cosine of the quantity specified on entry. This quantity must be in radians.

SPECIFIC NAMES: DCOS, QCOS, CCOS, CDCOS, CQCOS.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than or equal to $2^{18}\pi$ at single precision or $2^{50}\pi$ at double or quadruple precision, the result would be wholly inaccurate, and the program terminates with the following message:

```
COS ARG OUT OF RANGE
```

For complex arguments an error occurs if the modulus of the imaginary part is greater than the permitted maximum of 175.36644.

## COSH

DEFINITION: This set of functions gives the value of the hyperbolic cosine of the quantity specified on entry, i.e. $\frac{1}{2}(e^{x}+e^{-x})$.

SPECIFIC NAMES: DCOSH, QCOSH.

ERROR CONDITIONS: If the modulus of X is greater than or equal to 175.36644 the program will terminate with the following message:

```
COSH ARG OUT OF RANGE
```

## COTAN

DEFINITION: This set of functions gives the value of the cotangent of the quantity specified on entry. This quantity must be in radians.

SPECIFIC NAMES: DCOTAN, QCOTAN.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than or equal to $2^{18}\pi$ at single precision, or $2^{50}\pi$ at double or quadruple precision the result would be wholly inaccurate and the program terminates with the following message:

```
COT ARG OUT OF RANGE
```

If the argument passed to this function is so near an even multiple of $\pi$ that an overflow condition would exist, the program terminates with the following message:

```
COT ARG INAPPROPRIATE
```

## CPUTIME

← this is a real*8 function with no arguments

DEFINITION: This subroutine returns the cpu time in seconds which has been used by the current job since some arbitrary point prior to entry to the user program.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## CTIME

DEFINITION: This subroutine returns the present time of day as eight ISO characters in the form hh:mm:ss.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## DBLE

DEFINITION: This set of functions expresses in double precision form the quantity specified on entry.

SPECIFIC NAMES: DFLOAT, DBLEQ.

ERROR CONDITIONS: None.

## DIM

DEFINITION: This set of functions gives the positive difference of the quantities specified on entry.

The value of arg1 - the smaller of the two parameters is returned,
i.e. arg1 - min (arg1,arg2).

SPECIFIC NAMES: IDIM, DDIM, QDIM.

ERROR CONDITIONS: None.

## DPROD

DEFINITION: This function gives the double precision value of the product of the quantities specified on entry.

SPECIFIC NAME: DPROD.

ERROR CONDITIONS: None.

NOTE: DPROD is a specific name; there is no generic name.

## ERF

DEFINITION: This function gives the value of the error function of the quantity specified on entry.

The value of the error function

$$y = \frac{2}{\sqrt{\pi}} \int_0^x e^{-u^2} du$$

of the quantity specified by X is returned.

SPECIFIC NAMES: DERF, QERF.

ERROR CONDITIONS: None.

## ERFC

DEFINITION: This set of functions give the value of the complemented error function of the quantity specified on entry.

The value of the complemented error function

$$y = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-u^2} du$$

or

$y = 1 - \mathrm{erf}(X)$

of the quantity specified by X is returned.

SPECIFIC NAMES: DERFC, QERFC.

ERROR CONDITIONS: None.

## EXP

DEFINITION: This set of functions gives the value of 'e' raised to the power of the quantity specified on entry.

$e^{x+iy} = e^x.\cos(y) + i.e^x.\sin(y)$

SPECIFIC NAMES: DEXP, QEXP, CEXP, CDEXP, CQEXP.

ERROR CONDITIONS: If the argument of this function is greater than 174.673089 or less than -180.2181669, the program terminates with the following message:

```
EXP ARG OUT OF RANGE
```

For complex arguments an error occurs if the imaginary part is outside the range: $-2^{50}\pi <= X_i <= 2^{50}\pi$ .

## GAMMA

DEFINITION: This set of functions gives the value of the gamma function of the quantity specified on entry.

The value of the gamma function

$$\Gamma(x) = \int_0^\infty u^{x-1} e^{-u} du$$

of the quantity specified by X is returned.

SPECIFIC NAME: DGAMMA.

ERROR CONDITIONS: If X is less than $16^{-63}$ or if X is greater than 57.5744, the program terminates with the following message:

```
GAMMA ARG OUT OF RANGE
```

## HDATE

DEFINITION: This subroutine returns the current date as eight ISO characters in the form dd/mm/yy.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## HFIX

DEFINITION: This function converts the real expression specified on entry to an Integer*2 value.

SPECIFIC NAME: HFIX.

ERROR CONDITIONS: None.

NOTE: HFIX is a specific name; there is no generic name.

## ICHAR

DEFINITION: This function converts a single character to an integer value.

SPECIFIC NAME: ICHAR.

ERROR CONDITIONS: None.

NOTE: ICHAR is a specific name; there is no generic name.

## ICL9CECPUTIME

DEFINITION: This subroutine returns the cpu time in seconds which has been used by the current job since some arbitrary point prior to entry to the user program.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## ICL9CEDATE

DEFINITION: This subroutine returns the current date as eight EBCDIC characters in the form dd/mm/yy.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## ICL9CEDIAG

DEFINITION: This subroutine prints a routine traceback, so long as the compile time option NODIAG is not selected. It takes no parameters.

ERROR CONDITIONS: None

NOTE: This is an Edinburgh Fortran Option.

## ICL9CEFTRACE

DEFINITION: This subroutine may be called, with parameter N, to switch on and off the listing of labels and subroutine entries and exits during execution of a program. N takes one of the following values:

| | |
|---|---|
| N=0 | turns the tracing off |
| N=1 | traces subprogram entry and exit only |
| N=2 | traces subprogram entry and exit and all labels |

ERROR CONDITIONS: If N<0 or N>2 then the default value is 1.

NOTE: This is an Edinburgh Fortran Option.

## ICL9CELABELS

DEFINITION: A list of the last 32 labels, subprogram entries and exits may be produced by a call on this subroutine, so long as the compile time option NOLABELS is not selected.

ERROR CONDITIONS: None

NOTE: This is an Edinburgh Fortran Option.

## ICL9CETIME

DEFINITION: This subroutine returns the present time of day as eight EBCDIC characters in the form hh:mm:ss.

ERROR CONDITIONS: None.

NOTE: This is an Edinburgh Fortran Option.

## IMAG

DEFINITION: This function obtains the imaginary part y of the complex argument, X=x+iy, specified on entry.

SPECIFIC NAMES: AIMAG, DIMAG, QIMAG.

ERROR CONDITIONS: None.

## INDEX

DEFINITION: INDEX($a_1$,$a_2$) returns an integer value indicating the starting position within the character string $a_1$ of a substring identical to string $a_2$. If $a_2$ occurs more than once in $a_1$ the starting position of the first occurrence is returned. If $a_2$ does not occur in $a_1$ the value zero is returned. Also if LEN($a_1$) $<$ LEN($a_2$) then zero is returned.

SPECIFIC NAME: INDEX.

ERROR CONDITIONS: None.

NOTE: INDEX is a specific name; there is no generic name.

## INT

DEFINITION: This set of functions gives the value of the largest integer less than or equal to the absolute value of the quantity specified on entry. The value returned takes the same sign as the argument.

SPECIFIC NAMES: IFIX, IDINT, IQINT.

ERROR CONDITIONS: None.

## LEN

DEFINITION: This function returns the length of the character variable passed as the argument.

SPECIFIC NAME: LEN.

ERROR CONDITIONS: None.

NOTE: LEN is a specific name; there is no generic name.

## LGAMMA

DEFINITION: This set of functions gives the value of the log-gamma function of the quantity specified on entry.

The value of the logarithm to the base 'e' of the gamma function of the quantity specified by X, i.e. $\log_e \Gamma(X)$, is returned.

SPECIFIC NAMES: ALGAMA, DLGAMA.

ERROR CONDITIONS: If X is greater than or equal to $4.2913*10^{73}$ the program terminates with the following message:

```
LGAMMA ARG TOO LARGE
```

If X is less than or equal to zero the program terminates with the following message:

```
LGAMMA ARG NOT POSITIVE
```

## LOG

DEFINITION: This set of functions gives the value of the logarithm to the base 'e' of the quantity specified on entry.

For complex arguments, where $X=x+iy$, $\ln(x+iy)=a+ib$, where $a=\ln|x+iy|$ and b is the principal value of $\arctan(y/x)$.

SPECIFIC NAMES: ALOG, DLOG, QLOG, CLOG, CDLOG, CQLOG.

ERROR CONDITIONS: If X is zero, the program terminates with the following message:

```
LOG ARG ZERO
```

If X is negative, the program terminates with the following message:

```
LOG ARG NEGATIVE
```

## LOG10

DEFINITION: This set of functions gives the common logarithm to the base 10 of the quantity specified on entry.

SPECIFIC NAMES: ALOG10, DLOG10, QLOG10.

ERROR CONDITIONS: If X is zero, the program terminates with the following message:

```
LOG ARG ZERO
```

If X is negative the program terminates with the following message:

```
LOG ARG NEGATIVE
```

## MAX

DEFINITION: This set of functions gives the value of the largest argument specified on entry in the parameter list.

SPECIFIC NAMES: MAX0, AMAX1, DMAX1, QMAX1.

ERROR CONDITIONS: None.

## MAX1

DEFINITION: This function gives the integer value of the largest real argument specified on entry in the parameter list.

SPECIFIC NAME: MAX1.

ERROR CONDITIONS: None.

NOTE: MAX1 is a specific name; there is no generic name.

## MIN

DEFINITION: This set of functions gives the value of the smallest argument specified on entry in the parameter list.

SPECIFIC NAMES: MIN0, AMIN1, DMIN1, QMIN1.

ERROR CONDITIONS: None.

## MIN1

DEFINITION: This function gives the integer value of the smallest real argument specified on entry in the parameter list.

SPECIFIC NAME: MIN1.

ERROR CONDITIONS: None.

NOTE: MIN1 is a specific name; there is no generic name.

## MOD

DEFINITION: This set of functions gives the value of X(MOD Y) of the quantities specified on entry.

The value of Arg1 (mod Arg2) = Arg1 -[X]*Arg2 is returned, where [X] is the largest integer whose magnitude does not exceed the magnitude of Arg1/Arg2. The sign of the integer is the same as the sign of Arg1/Arg2.

SPECIFIC NAMES: AMOD, DMOD, QMOD.

ERROR CONDITIONS: If a division by zero is attempted the program will terminate with the following message:

```
ZERO DIVIDE
```

## NINT

DEFINITION: This set of functions gives the nearest integer to the quantity specified on entry.

i.e. INT(X+0.5) if $X \geqslant 0$
INT(X-0.5) if $X < 0$

SPECIFIC NAME: IDNINT.

ERROR CONDITIONS: None.

## QEXT

DEFINITION: This function expresses in Real*16 form the quantity specified on entry.

SPECIFIC NAME: QEXTD.

ERROR CONDITIONS: None.

## REAL

DEFINITION: This function obtains the real part x of the complex argument, $X=x+iy$, specified on entry, or converts to real the integer argument.

SPECIFIC NAMES: FLOAT, SNGL, DREAL, QREAL.

ERROR CONDITIONS: None.

## SIGN

DEFINITION: This set of functions transfers the sign of the second argument to the modulus of the first.

SPECIFIC NAMES: ISIGN, DSIGN, QSIGN.

ERROR CONDITIONS: None.

## SIN

DEFINITION: This set of functions gives the value of the sine of the quantity specified on entry. This quantity must be in radians.

SPECIFIC NAMES: DSIN, QSIN, CSIN, CDSIN, CQSIN.

ERROR CONDITIONS: If the modulus of the argument passed to this function is greater than or equal to $2^{18}\pi$ at single precision, or $2^{50}\pi$ at double or quadruple precision, the result would be wholly inaccurate and the program terminates with the following message:

```
SIN ARG OUT OF RANGE
```

For complex arguments an error occurs if the modulus of the imaginary part is greater than the permitted maximum of 175.36644.

## SINH

DEFINITION: This set of functions gives the value of the hyperbolic sine of the quantity specified on entry.

SPECIFIC NAMES: DSINH, QSINH.

ERROR CONDITIONS: If the modulus of X is greater than 175.36644 the program will terminate with the following message:

```
SINH ARG OUT OF RANGE
```

## SQRT

DEFINITION: This set of functions gives the value of the positive square root of the quantity specified on entry.

SPECIFIC NAMES: DSQRT, QSQRT, CSQRT, CDSQRT, CQSQRT.

ERROR CONDITIONS: If a negative argument is passed to this routine the program terminates with the following message:

```
SQRT ARG NEGATIVE
```

## TAN

DEFINITION: This set of functions gives the value of the tangent of the quantity specified on entry. This quantity must be in radians.

SPECIFIC NAMES: DTAN, QTAN.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than or equal to $2^{18}\pi$ at single precision or $2^{50}\pi$ at double or quadruple precision the result would be wholly inaccurate and the program terminates with the following message:

```
TAN ARG OUT OF RANGE
```

If the argument passed to this function is so near an odd multiple of $\frac{\pi}{2}$ that an overflow condition would occur, the program terminates with the message:

```
TAN ARG INAPPROPRIATE
```

## TANH

DEFINITION: This set of functions gives the value of the hyperbolic tangent of the quantity specified on entry.

SPECIFIC NAMES: DTANH, QTANH.

ERROR CONDITIONS: None.

# 3

# IMP

## 3.1 INTRINSIC, IMPLICIT, EXPLICIT Functions in IMP

Routines, functions and maps are divided into three classes INTRINSIC, IMPLICIT and EXPLICIT. Items in the first two classes can be used without explicit declaration since their names and characteristics are known to the compiler. For INTRINSIC functions the compiler plants in-line code while for IMPLICIT functions the compiler generates an appropriate specification. Items in the EXPLICIT class, however, must be specified before they are used. The specification provides information to the compiler as to the name and parameter list of the routine, function or map. It also causes the compiler to generate an entry in a table to ensure that the necessary module is loaded when the program is run. It is most important to type the specification accurately, and in particular the number and types of parameter must be correct (the names of parameters used are not significant).

The specification of items in the EXPLICIT class are included in the item descriptions, which are given in alphabetical order, after the table following.

## 3.2 REFERENCE TABLE AND INDEX FOR IMP LIBRARY

| NAME | TYPE | CLASS | PARAMETERS | PAGE NO. |
|---|---|---|---|---|
| ADD MATRIX | routine | EXPLICIT | longrealarrayname A,B,C, integer I,J | 3-6 |
| ADDR | integerfn | INTRINSIC | name I | 3-6 |
| ARCCOS | longrealfn | IMPLICIT | longreal A | 3-6 |
| ARCSIN | longrealfn | IMPLICIT | longreal A | 3-7 |
| ARCTAN | longrealfn | IMPLICIT | longreal A,B | 3-7 |
| ARRAY | arraymap | INTRINSIC | integer I, arrayname J or<br>integer I, arrayformat J | 3-8 |
| BITS | integerfn | EXPLICIT | integer I | 3-8 |
| BYTE INTEGER | byteintegermap | INTRINSIC | integer I | 3-9 |
| CHARNO | byteintegermap | INTRINSIC | stringname S, integer I | 3-9 |
| CLOSE DA | routine | EXPLICIT | integer I | 3-10 |
| CLOSE SQ | routine | EXPLICIT | integer I | 3-10 |
| CLOSE STREAM | routine | IMPLICIT | integer I | 3-10 |
| COPY MATRIX | routine | EXPLICIT | longrealarrayname A,B, integer I,J | 3-11 |
| COS | longrealfn | IMPLICIT | longreal A | 3-11 |
| COT | longrealfn | IMPLICIT | longreal A | 3-11 |
| CPUTIME | longrealfn | EXPLICIT | None | 3-12 |
| DATE | stringfn | EXPLICIT | None | 3-12 |
| DET | longrealfn | EXPLICIT | longrealarrayname A, integer I | 3-13 |
| DIV MATRIX | routine | EXPLICIT | longrealarrayname A,B,<br>integer I,J, longrealname C | 3-13 |
| ERFN | longrealfn | EXPLICIT | longreal A | 3-14 |
| ERFNC | longrealfn | EXPLICIT | longreal A | 3-14 |
| EXP | longrealfn | IMPLICIT | longreal A | 3-15 |
| EXP TEN | longrealfn | EXPLICIT | longreal A | 3-15 |
| FRAC PT | longrealfn | INTRINSIC | longreal A | 3-15 |
| FROM STRING | stringfn | IMPLICIT | stringname S, integer I,J | 3-16 |
| GAMMAFN | longrealfn | EXPLICIT | longreal A | 3-16 |
| HYPCOS | longrealfn | EXPLICIT | longreal A | 3-17 |
| HYPSIN | longrealfn | EXPLICIT | longreal A | 3-17 |
| HYPTAN | longrealfn | EXPLICIT | longreal A | 3-18 |
| IMOD | integerfn | INTRINSIC | integer I | 3-18 |
| INSTREAM | integerfn | EXPLICIT | None | 3-18 |
| INT | integerfn | INTRINSIC | longreal A | 3-19 |

| NAME | TYPE | CLASS | PARAMETERS | PAGE NO. |
|---|---|---|---|---|
| INTEGER | integermap | INTRINSIC | integer I | 3-19 |
| INT PT | integerfn | INTRINSIC | longreal A | 3-20 |
| INVERT | routine | EXPLICIT | longrealarrayname A,B, integer I, longrealname J | 3-20 |
| ISO CARD | routine | EXPLICIT | byteintegerarrayname K | 3-21 |
| LENGTH | byteintegermap | INTRINSIC | stringname S | 3-21 |
| LENGTHENI | longintegerfn | IMPLICIT | integer I | 3-21 |
| LENGTHENR | longlongrealfn | IMPLICIT | longreal A | 3-22 |
| LINT | longintegerfn | IMPLICIT | longlongreal A | 3-22 |
| LINTPT | longintegerfn | IMPLICIT | longlongreal A | 3-22 |
| LOG | longrealfn | IMPLICIT | longreal A | 3-23 |
| LOGGAMMA | longrealfn | EXPLICIT | longreal A | 3-23 |
| LOGTEN | longrealfn | EXPLICIT | longreal A | 3-24 |
| LONG INTEGER | longintegermap | INTRINSIC | integer I | 3-24 |
| LONG LONG REAL | longlongrealmap | INTRINSIC | integer I | 3-24 |
| LONG REAL | longrealmap | INTRINSIC | integer I | 3-25 |
| LRANDOM | longrealfn | EXPLICIT | integername I, integer N | 3-25 |
| MOD | longrealfn | INTRINSIC | longreal A | 3-26 |
| MULT MATRIX | routine | EXPLICIT | longrealarrayname A,B,C, integer I,J,K | 3-26 |
| MULT TR MATRIX | routine | EXPLICIT | longrealarrayname A,B,C, integer I,J,K | 3-27 |
| NEWLINE | routine | INTRINSIC | None | 3-27 |
| NEWLINES | routine | INTRINSIC | integer I | 3-28 |
| NEWPAGE | routine | INTRINSIC | None | 3-28 |
| NEXT CH | integerfn | INTRINSIC | None | 3-29 |
| NEXT ITEM | stringfn | INTRINSIC | None | 3-29 |
| NEXT SYMBOL | integerfn | INTRINSIC | None | 3-30 |
| NL | integerfn | INTRINSIC | None | 3-30 |
| NULL | routine | EXPLICIT | longrealarrayname A, integer I,J | 3-31 |
| OPEN DA | routine | EXPLICIT | integer I | 3-31 |
| OPEN SQ | routine | EXPLICIT | integer I | 3-32 |
| OUTPOS | integerfn | EXPLICIT | None | 3-32 |
| OUT STREAM | integerfn | EXPLICIT | None | 3-32 |
| PI | longrealfn | INTRINSIC | None | 3-33 |
| PRINT | routine | IMPLICIT | longreal A, integer I,J | 3-33 |

| NAME | TYPE | CLASS | PARAMETERS | PAGE NO. |
|---|---|---|---|---|
| PRINT CH | routine | INTRINSIC | integer I | 3-34 |
| PRINT FL | routine | IMPLICIT | longreal A, integer I | 3-34 |
| PRINT STRING | routine | INTRINSIC | string S | 3-34 |
| PRINT SYMBOL | routine | INTRINSIC | integer I | 3-35 |
| RADIUS | longrealfn | IMPLICIT | longreal A,B | 3-35 |
| RANDOM | realfn | EXPLICIT | integername I, integer K | 3-36 |
| READ | routine | IMPLICIT | name A | 3-37 |
| READ CH | routine | INTRINSIC | name I | 3-37 |
| READ DA | routine | EXPLICIT | integer I, integername J, name K,L | 3-38 |
| READ ITEM | routine | INTRINSIC | stringname S | 3-39 |
| READ LSQ | routine | EXPLICIT | integer I, name N,M, integername J | 3-39 |
| READ SQ | routine | EXPLICIT | integer I, name J,K | 3-40 |
| READ STRING | routine | IMPLICIT | stringname S | 3-41 |
| READ SYMBOL | routine | INTRINSIC | name I | 3-42 |
| REAL | realmap | INTRINSIC | integer I | 3-42 |
| RECORD | recordmap | INTRINSIC | integer I | 3-43 |
| SELECT INPUT | routine | INTRINSIC | integer I | 3-43 |
| SELECT OUTPUT | routine | INTRINSIC | integer I | 3-44 |
| SET MARGINS | routine | IMPLICIT | integer I,J,K | 3-45 |
| SHIFTC | integerfn | EXPLICIT | integer I,J | 3-46 |
| SHORTENI | integerfn | IMPLICIT | longinteger I | 3-46 |
| SHORTENR | longrealfn | IMPLICIT | longlongreal A | 3-46 |
| SIN | longrealfn | IMPLICIT | longreal A | 3-47 |
| SKIP SYMBOL | routine | INTRINSIC | None | 3-47 |
| SOLVE LN EQ | routine | EXPLICIT | longrealarrayname A,B, integer I, longrealname C | 3-48 |
| SPACE | routine | INTRINSIC | None | 3-48 |
| SPACES | routine | INTRINSIC | integer I | 3-49 |
| SQRT | longrealfn | IMPLICIT | longreal A | 3-49 |
| STRING | stringmap | INTRINSIC | integer I | 3-50 |
| SUB MATRIX | routine | EXPLICIT | longrealarrayname A,B,C, integer I,J | 3-50 |
| TAN | longrealfn | IMPLICIT | longreal A | 3-51 |
| TIME | stringfn | EXPLICIT | None | 3-51 |
| TO STRING | stringfn | INTRINSIC | integer I | 3-51 |

| NAME | TYPE | CLASS | PARAMETERS | PAGE NO. |
|---|---|---|---|---|
| TRANS MATRIX | routine | EXPLICIT | longrealarrayname A,B, integer I,J | 3-52 |
| UNIT | routine | EXPLICIT | longrealarrayname A, integer I | 3-52 |
| WRITE | routine | INTRINSIC | integer I,J | 3-53 |
| WRITE DA | routine | EXPLICIT | integer I, integername J, name K,L | 3-54 |
| WRITE SQ | routine | EXPLICIT | integer I, name J,K | 3-55 |

## ADD MATRIX

DEFINITION: This routine adds two NxM matrices.

SPEC: externalroutinespec ADD MATRIX (longrealarrayname A,B,C, integer N,M)

CALL: ADD MATRIX (A,B,C,N,M)

| | |
|---|---|
| A,B,C | are the names of three longreal two-dimensional arrays. Array A contains the matrix sum of matrices in arrays B and C. Arrays B and C contain the matrices to be added: arrays B and C are not corrupted. |
| N,M | are the integer expressions whose values give upper bounds for the dimensions of A, B and C, the lower bounds being taken to be 1. |

ERROR CONDITIONS: If the array bounds of any of the matrices are zero or negative, the program terminates with the message:

```
MATRIX BOUND ZERO OR NEGATIVE
```

## ADDR

DEFINITION: This integer function gives the absolute address of the name type parameter specified on entry.

SPEC: Not required

CALL: ADDR(X)

| | |
|---|---|
| X | is the name of a variable whose address is to be returned. |

ERROR CONDITIONS: None.

## ARCCOS

DEFINITION: This longreal function finds the value of the angle whose cosine is specified on entry.

SPEC: Not required

CALL: ARCCOS(X)

| | |
|---|---|
| X | is a real or longreal expression. |

The value is given in radians, and lies in the range 0 to $+\pi$

ERROR CONDITIONS: If x is not in the range -1 <= x <= 1, the program terminates with the message:

```
ARCCOS ARG OUT OF RANGE
```

## ARCSIN

DEFINITION: This longreal function finds the value of the angle whose sine is specified on entry.

SPEC: Not required

CALL: ARCSIN(X)

X is a real or longreal expression.

The value is given in radians, and lies in the range $-\pi/2$ to $+\pi/2$.

ERROR CONDITIONS: If x is not in the range -1 <= x <= 1, the program terminates with the message:

```
ARCSIN ARG OUT OF RANGE
```

## ARCTAN

DEFINITION: This longreal function gives the value of the angle whose tangent is specified on entry.

SPEC: Not required

CALL: ARCTAN(X,Y)

X,Y are real or longreal expressions.

The value, in radians, of the angle whose tangent is specified by 'y/x' is returned. This value lies in the range $-\pi$ to $+\pi$, and is in the first or fourth quadrant if $x > 0$, and in the second or third quandrant if $x < 0$.

ERROR CONDITIONS: If the parameters passed to this routine are both zero the program terminates with the message:

```
ARCTAN ARGS ZERO
```

## ARRAY

DEFINITION: This map can be used to map an array onto an area specified by the address of its start.

SPEC: Not required

CALL: ARRAY(I,J)

| | |
|---|---|
| I | is an integer expression specifying an address |
| J | is the name of an array or an arrayformat |

The arrayformat statement is used to describe the characteristics of the array which is being mapped. As an alternative to using the name of an arrayformat for the second parameter, the name of another array can be used, if one with suitable characteristics has been defined in the program.

In the following example the two-dimensional array ATWO is mapped onto an array AONE, which is declared as a one-dimensional array:

```
integerarray AONE (1:10000)
integerarrayname ATWO
integerarrayformat AFORM (1:100,1:100) ;  !This statement describes the
                                          !characteristics of the array ATWO
   ATWO==ARRAY(ADDR(AONE(1)),AFORM)
   ATWO(27,27)=928
```

ERROR CONDITIONS: None.

## BITS

DEFINITION: This integer function finds the number of non-zero bits in the binary representation of a specified integer.

SPEC: externalintegerfnspec BITS (integer N)

CALL: BITS(N)

| | |
|---|---|
| N | is the integer expression whose binary representation is to be examined. |

ERROR CONDITIONS: None

## BYTE INTEGER

DEFINITION: This map enables the user to access a particular byte location whose address is specified by the parameter.

SPEC: Not required

CALL: BYTE INTEGER(N)

N is an integer expression giving the absolute address in core of the required location.

This map may be used to access a byte integer location as though it held a byteinteger value, or to store a byteinteger value at the specified address:

```
integer I
byteinteger J
I=X'12345678'  ;                    !I is an integer containing a 4 byte binary pattern
J=BYTEINTEGER (ADDR(I)+1)  ;        !J has the value X'34' and I is unchanged
BYTEINTEGER (ADDR(I)+3)=X'9A' ;     !I now contains the binary pattern X'1234569A' and
                                    !J is unchanged
```

ERROR CONDITIONS: None.

## CHARNO

DEFINITION: This map enables the user to access a character in a specified position within a string.

SPEC: Not required

CALL: CHARNO(S,N)

S is the name of the string variable containing the string.
N is an integer expression indicating which character of the string is to be accessed.

ERROR CONDITIONS: In checking mode a bound check occurs but otherwise the result is undefined unless 1 <= n <= LENGTH(S).

## CLOSEDA

DEFINITION: This routine closes the direct access file on the channel specified on entry.

SPEC: <u>externalroutinespec</u> CLOSEDA (<u>integer</u> N)

CALL: CLOSEDA(N)

N is an integer expression which specifies the channel number and must be in the the range allowed by the operating system being used.

The file is closed and may not be used again without re-opening.

ERROR CONDITIONS: If an attempt is made to close a file twice or to close a file which has been neither opened nor defined, the program terminates with an appropriate message.

## CLOSESQ

DEFINITION: The routine closes the sequential file on the channel specified on entry.

SPEC: <u>externalroutinespec</u> CLOSESQ (<u>integer</u> M)

CALL: CLOSESQ(M)

M is an integer expression which specifies the channel number and must be in the range allowed by the operating system being used.

The file is closed and may not be used again without re-opening.

ERROR CONDITIONS: If an attempt is made to close a file twice, or to close a file which has been neither opened nor defined, the program terminates with an appropriate message.

## CLOSE STREAM

DEFINITION: This routine closes a user-defined stream.

SPEC: Not required

CALL: CLOSE STREAM(N)

N is an integer expression which specifies the stream to be closed.

The stream n is closed such that on the next call of either SELECT INPUT or SELECT OUTPUT for that stream, the stream is reset to the start. Note that n must not be the current input or output stream.

ERROR CONDITIONS: If the stream is not defined on entry to the routine, if an attempt is made to close a stream not in the range allowed by the operating system being used, or if n is a current stream, then the program terminates with an appropriate message.

## COPY MATRIX

DEFINITION: An NxM matrix is copied from one two-dimensional array onto another.

SPEC: <u>externalroutinespec</u> COPY MATRIX(<u>longrealarrayname</u> A,B, <u>integer</u> N,M)

CALL: COPY MATRIX (A,B,N,M)

| | |
|---|---|
| A,B | are the names of two longreal two-dimensional arrays. Array A contains a matrix identical to the matrix contained in B. Array B contains the rectangular matrix which is to be copied: array B is unchanged. |
| N,M | are the integer expressions whose values give the upper limits for the dimensions of the given arrays, the lower limits being taken to be 1. |

ERROR CONDITIONS: If n or m is negative or zero, the program terminates with the message:

MATRIX BOUND ZERO OR NEGATIVE

## COS

DEFINITION: This longreal function gives the value of the cosine of the quantity specified on entry. This quantity must be in radians.

SPEC: Not required

CALL: COS(X)

X is a real or longreal expression.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than $3.53 \times 10^{15}$ the result would be wholly inaccurate and the program terminates with the following message:

COS ARG OUT OF RANGE

## COT

DEFINITION: This longreal function gives the value of the cotangent of the quantity specified on entry. This quantity must be in radians.

SPEC: Not required

CALL: COT(X)

X is a real or longreal expression.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than or equal to $10^{15}$ the result would be wholly inaccurate and the program terminates with the message:

COT ARG OUT OF RANGE

If the argument passed to this function is near 0 or a multiple of $\pi$, the program terminates with the message:

COT ARG INAPPROPRIATE

## CPUTIME

DEFINITION: This longreal function gives the total CPU time used from an arbitrary starting point.

SPEC: externallongrealfnspec CPUTIME

CALL: CPUTIME

No parameters

The total CPU time used since an arbitrary starting point is returned as a positive long real number having seconds as its unit of time. Note that there is a small CPU time overhead in calling the function itself.

ERROR CONDITIONS: None.

## DATE

DEFINITION: This string function yields the current date.

SPEC: externalstringfnspec DATE

CALL: DATE

No parameters.

The date is returned in the form dd/mm/yy, and may be printed out using PRINTSTRING.

ERROR CONDITIONS: None.

## DET

DEFINITION: This longreal function evaluates the determinant of an NxN matrix. The method of Gaussian elimination with partial pivoting is used. This is implemented using the routine SOLVE LN EQ.

SPEC: <u>externallongrealfnspec</u> DET (<u>longrealarrayname</u> A, <u>integer</u> N)

CALL: DET (A,N)

| | |
|---|---|
| A | is the name of a longreal two-dimensional array containing the square matrix whose determinant is required. Note that the contents of this matrix are destroyed. |
| N | is the integer expression whose value gives the upper limit for the dimensions of A, the lower limits taken to be 1. |

The value of the determinant is returned. Note that large errors may occur if the matrix is ill-conditioned, and if this is suspected, perturbed matrices should be examined.

ERROR CONDITIONS: If the matrix bound n is zero or negative, the program terminates with the message:

```
MATRIX BOUND ZERO OR NEGATIVE
```

## DIV MATRIX

DEFINITION: Given an NxN matrix B and an NxM matrix A, the routine generates INV(B).A, an NxM matrix. Both B and A are destroyed.

SPEC: <u>externalroutinespec</u> DIV MATRIX (<u>longrealarrayname</u> A,B, <u>integer</u> N,M, <u>longrealname</u> DET)

CALL: DIV MATRIX (A,B,N,M,DET)

| | |
|---|---|
| A,B | are the names of two two-dimensional longreal arrays. Array A contains rectangular matrix A on input. Array B contains square matrix B on input. The result of the division is placed in array A and matrix B is destroyed. |
| N,M | are the integer expressions whose values give the upper limits for the dimensions of the given arrays, the lower limits being taken to be 1. |
| DET | is the name of a longreal variable which is set by the routine to the value of the determinant of the matrix in array B. |

Note that large errors may occur if the matrix B is ill-conditioned, and if this is suspected, perturbed matrices should be examined.

ERROR CONDITIONS: If N or M is less than or equal to zero, the program terminates with the message:

```
DIV MATRIX DATA FAULT N = X
```

where X is the invalid value N or M.

## ERFN

DEFINITION: This real or longreal function gives the value of the error function of the quantity specified on entry.

SPEC: <u>externallongrealfnspec</u> ERFN(<u>longreal</u> X)

CALL: ERFN(X)

X is a real or longreal expression.

The real or longreal value of the error function

$$y = \frac{2}{\sqrt{\pi}} \int_0^\infty e^{-u} du$$

of the quantity specified by X is returned.

ERROR CONDITIONS: None.

## ERFNC

DEFINITION: This real or longreal function gives the value of the complemented error function of the quantity specified on entry.

SPEC: <u>externallongrealfnspec</u> ERFNC (<u>longreal</u> X)

CALL: ERFNC(X)

X is a real or longreal expression

The real or longreal value of the complemented error function

$$y = \frac{2}{\sqrt{\pi}} \int_x^\infty e^{-u^2} du$$

or

$y = 1 - \text{erfn}(X)$

of the quantity specified by X is returned.

ERROR CONDITIONS: None.

## EXP

DEFINITION: This real or longreal function gives the value of 'e' raised to the power of the quantity specified on entry.

SPEC: Not required

CALL: EXP(X)

X is a real or longreal expression.

If the argument of this function is less than -180.218 then zero is returned.

ERROR CONDITIONS: If the argument of this function is greater than 174.673, the program terminates with the message:

```
EXP ARG OUT OF RANGE
```

## EXPTEN

DEFINITION: This real or longreal function gives the value of 10 raised to the power of the quantity specified on entry.

SPEC: <u>externallongrealfnspec</u> EXPTEN (<u>longreal</u> X)

CALL: EXPTEN(X)

X is a real or longreal expression.

ERROR CONDITIONS: If the argument of this function is greater than 75.0 then the program terminates with the message:

```
EXP ARG OUT OF RANGE
```

## FRAC PT

DEFINITION: This longreal function gives the value of the fractional part of the longreal quantity specified on entry.

SPEC: Not required

CALL: FRAC PT(X)

X is a longreal expression of which the fractional part is to be returned.

The fractional part of x is returned as a long real. If mod(x) exceeds $2^{55}-1$, the value returned is always zero. The fractional part is always calculated as

x - (largest integer less than or equal to x);

```
e.g. FRACPT(4.6)=0.6
     FRACPT(-4.6)=0.4
```

ERROR CONDITIONS: None.

## FROM STRING

DEFINITION: This string function extracts characters from a string.

SPEC: Not required

CALL: FROM STRING (S,I,J)

S  is the name (of type string) of the location in which the string is stored.

I,J  are integer expressions giving the lower and upper boundaries (inclusive) of the character string to be extracted.

The result is a string of characters extracted from the string contained in S (from I to J inclusive): s is unchanged.

ERROR CONDITIONS: Unless 1 <= i <= j <= LENGTH(S), the program terminates with the message:

```
STRING INSIDE OUT
```

## GAMMAFN

DEFINITION: This longreal function gives the value of the gamma function of the quantity specified on entry.

SPEC: externallongrealfnspec GAMMAFN(longreal X)

CALL: GAMMAFN(X)

X  is a real or longreal expression

The real or longreal value of the gamma function

$$\Gamma(x) = \int_0^\infty u^{x-1}e^{-u}du$$

of the quantity specified by X is returned.

ERROR CONDITIONS: If mod(x) is less than $1*10^{-11}$ the program will terminate with the message:

```
LGAMMA ARG NOT POSITIVE
```

If x is greater than 57 the program terminates with the message:

```
LGAMMA ARG TOO LARGE
```

## HYPCOS

DEFINITION: This real or longreal function gives the value of the hyperbolic cosine of the quantity specified on entry.

SPEC: externallongrealfnspec HYPCOS(longreal X)

CALL: HYPCOS(X)

X is a real or longreal expression.

The real or longreal value of the hyperbolic cosine

$y = \frac{1}{2}(e^x + e^{-x})$

of the quantity specified by X is returned.

ERROR CONDITIONS: If the modulus of x is greater than or equal to 172.694 the program will terminate with the message:

HYPCOS ARG OUT OF RANGE

## HYPSIN

DEFINITION: This real or longreal function gives the values of the hyperbolic sine of the quantity specified on entry.

SPEC: externallongrealfnspec HYPSIN(longreal X)

CALL: HYPSIN(X)

X is a real or longreal expression.

The real or longreal value of the hyperbolic sine

$y = \frac{1}{2}(e^x - e^{-x})$

of the quantity specified by X is returned.

ERROR CONDITIONS: If the modulus of x is greater than or equal to 172.694 the program will terminate with the message:

HYPSIN ARG OUT OF RANGE

## HYPTAN

DEFINITION: This real or longreal function gives the value of the hyperbolic tangent of the quantity specified on entry.

SPEC: <u>externallongrealfnspec</u> HYPTAN(<u>longreal</u> X)

CALL: HYPTAN(X)

X is a real or longreal expression.

The real or longreal value of the hyperbolic tangent

$$y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

of the quantity specified by X is returned.

ERROR CONDITIONS: None.

## IMOD

DEFINITION: This integer function gives the value of the modulus of the integer quantity specified on entry.

SPEC: Not required.

CALL: IMOD(N)

N is an integer expression.

ERROR CONDITIONS: Integer overflow occurs if an attempt is made to take the modulus of the largest negative integer (-2147483648).

## INSTREAM

DEFINITION: This integer function returns the number of the stream currently selected for input.

SPEC: <u>externalintegerfnspec</u> INSTREAM

CALL: INSTREAM

No parameters.

ERROR CONDITIONS: None

## INT

DEFINITION: This integer function gives the value of the nearest integer to the quantity specified on entry.

SPEC: Not required.

CALL: INT(X)

X is a longreal expression.

The value of the nearest integer to x is returned.

ERROR CONDITIONS: In checking mode if (x+ 0.5) is outwith the range of a 32-bit integer the program terminates with the message: ?

INT PT TOO LARGE

This error is trappable as event 1/7.

## INTEGER

DEFINITION: This map enables the user to access a particular 4-byte location whose address is specified by the parameter.

SPEC: Not required.

CALL: INTEGER(N)

N is an integer expression giving the absolute address in core of the required location.

This map may be used to access the contents of a correctly aligned 4-byte location, as though it held an <u>integer</u> variable, or to store an integer value at the specified address:

```
integer I
ownbyteintegerarray A(1:4) = 'A','B','C','D'
I = INTEGER(ADDR(A(1)));              !I now contains M'ABCD'
INTEGER(ADDR(A(1))) = 0;              !All elements of A now contain 0
```

ERROR CONDITIONS: If the specified location is not correctly full-word aligned, an address error occurs when the routine attempts to access it.

## INT PT

DEFINITION: This integer function gives the value of the integral part of the quantity specified on entry.

SPEC: Not required.

CALL: INT PT(X)

X is a real or longreal expression.

The integral part of x is returned.
The integral part is the integer that is less than or equal to the expression.
Hence INT PT (-3.6) is -4.

ERROR CONDITIONS: In checking mode if (x + 0.5) is outwith the range of a 32-bit integer the program terminates with the message:

```
INT PT TOO LARGE
```

This error is trappable as event 1/7.

## INVERT

DEFINITION: This routine inverts an NxN matrix, making use of the routines DIV MATRIX and UNIT. The method of Gaussian elimination with partial pivoting is used. The given matrix is destroyed.

SPEC: externalroutinespec INVERT (longrealarrayname A,B, integer N, longrealname DET)

CALL: INVERT (A,B,N,DET)

| | |
|---|---|
| A,B | are the names of two longreal two-dimensional arrays. Array B contains the matrix to be inverted. The required inverse is placed in A and the input matrix is destroyed. |
| N | is the integer expression whose value gives an upper limit for the dimensions of the given arrays, the lower bounds being taken to be 1. |
| DET | is the name of a longreal variable which is set to the value of the determinant of the matrix in array B. |

Note that large errors may occur if the matrix is ill-conditioned, and if this is suspected, perturbed matrices should be examined.

ERROR CONDITIONS: If N is less than, or equal to zero, the program terminates with the message:

```
INVERT DATA FAULT N = n
```

## ISO CARD

DEFINITION: This routine reads the next card or next card image from the currently selected input stream, and fills the 80-byte array specified on entry starting at element 1 of the array.

SPEC: externalroutinespec ISO CARD (byteintegerarrayname A)

CALL: ISO CARD (A)

A is the name of a byte integer array.

The array A contains the Edinburgh ISO equivalents of the card punching.

Note that valid extended Hollerith characters which have no ISO equivalent will be interpreted as SUB(X'1A'). Punchings which lie outside the 256 character extended Hollerith code will result in card rejection by the hardware.
When reading from an EMAS file records of less than 80 bytes are space filled; records greater than 80 bytes are truncated and the excess lost.

ERROR CONDITIONS: None.

## LENGTH

DEFINITION: This map accesses the location containing the current length of a string.

SPEC: Not required.

CALL: LENGTH(S)

S is the name (of type string) of the location containing the string whose length byte is to be accessed.

This map enables the number of characters in a string to be measured or adjusted without knowing the physical layout of the string.

Note if the length of the string is increased the characters 'added' to the string will be undefined.

ERROR CONDITIONS: None.

## LENGTHENI

DEFINITION: This longinteger function extends an operand in an expression from type integer to longinteger

SPEC: Not required.

CALL: LENGTHEN(I)

I is an integer expression.

ERROR CONDITIONS: None

## LENGTHENR

DEFINITION: This longlongreal function extends an operand in an expression from type longreal to longlongreal

SPEC Not required.

CALL: LENGTHENR(A)

A is a longreal expression.

ERROR CONDITIONS: None.

## LINT

DEFINITION: This longinteger function gives the value of the integer nearest to the value specified on entry.

SPEC: Not required.

CALL: LINT(A)

A is a longlongreal expression.

ERROR CONDITIONS: Overflow will occur if (A + 0.5) is outside the range of a longinteger.

## LINTPT

DEFINITION: This longinteger function gives the value of the largest integer less than or equal to the value specified on entry.

SPEC: Not required.

CALL: LINTPT(A)

A is a longlongreal expression.

ERROR CONDITIONS: Overflow will occur if (A + 0.5) is outside the range of a longinteger.

## LOG

DEFINITION: This longreal function gives the value of the logarithm to the base 'e' of the quantity specified on entry.

SPEC: Not required.

CALL: LOG(X)

X is a real or longreal expression.

ERROR CONDITIONS: If x is negative the program terminates with the message:

LOG ARG NEGATIVE

If x is zero the program terminates with the message:

LOG ARG ZERO

## LOGGAMMA

DEFINITION: This longreal function gives the value of the log-gamma function of the quantity specified on entry.

SPEC: externallongrealfnspec LOGGAMMA(longreal X)

CALL: LOGGAMMA(X)

X is a real or longreal expression.

The real or longreal value of the logarithm to the base 'e' of the gamma function of the quantity specified by X, i.e. $\log_e \Gamma(x)$ is returned.

ERROR CONDITIONS: If x is less than or equal to zero, the program terminates with the message:

LGAMMA ARG NOT POSITIVE

If x is greater than or equal to $4.2913*10^{73}$ the program terminates with the message:

LGAMMA ARG TOO LARGE

## LOGTEN

DEFINITION: This real or longreal function gives the common logarithm to the base 10 of the quantity specified on entry.

SPEC: <u>externallongrealfnspec</u> LOGTEN (<u>longreal</u> X)

CALL: LOGTEN(X)

X is a real or longreal expression.

ERROR CONDITIONS: If x is negative the program terminates with the message:

LOG ARG NEGATIVE

If x is zero the program terminates with the message:

LOG ARG NEGATIVE

## LONG INTEGER

DEFINITION: This longinteger map enables the user to access a particular 8-byte location whose address is specified on entry.

SPEC: Not required.

CALL: LONGINTEGER(I)

I is an integer expression giving the absolute address in core of the required location.

ERROR CONDITIONS: If the specified address is not double-word aligned or refers to a protected area of core then the effect is undefined.

## LONG LONG REAL

DEFINITION: This longlongreal map enables the user to access a particular 16-byte location whose address is specified by the parameter.

SPEC: Not required.

CALL: LONGLONGREAL(I)

I Is an integer expression giving the address of the location to be accessed.

ERROR CONDITIONS: If the address is not double-word aligned or refers to a protected area of core then the effect is undefined.

## LONG REAL

DEFINITION: This map enables the user to access a particular long real location whose address is specified as the parameter.

SPEC: Not required.

CALL: LONG REAL (N)

N is an integer expression giving the address of the location to be accessed.

This map may be used to access the contents of a correctly aligned location as though it were a longreal variable, or to store a longreal value at the specified address:

```
integerarray A(1:2)
longreal R
R = 3.1415926535897932;       !PI, double precision
LONGREAL(ADDR(A(1))) = R;     !array A now contains the 64 bit pattern in R
A(2) = 0
R = LONGREAL(ADDR(A(1)));     !R now contains PI, single precision.
```

Note that LONG REAL is always double length and is not affected by realsnormal.

ERROR CONDITIONS: If the specified location is not correctly double word aligned the effect when the map attempts to access it is undefined.

## LRANDOM

DEFINITION: This longreal function produces a set of random numbers either in a rectangular or Gaussian distribution.

SPEC: externallongrealfnspec LRANDOM(integername I, integer N)

CALL: LRANDOM(I,N)

I is the name of an integer variable which must be initialised to any odd integer. Note that a value of I < 50001 may produce a sequence which is similar in the first few terms.

N is an integer expression whose value controls the type of distribution:
N = 1 gives a rectangular distribution,
n > 1 gives a Gaussian distribution with a mean value of n/2 and a standard deviation of $\sqrt{\frac{n}{12}}$. For a Gaussian distribution a value of n greater than 10 is recommended.

A longreal number in the range 0-n is returned. Note that I contains an integer random sequence in the range 0-($2^{31}$-1).

Users may wish to scale the distribution. To produce Z with Gaussian distribution, mean X, and standard deviation S, call LRANDOM with N = 12 and scale thus:-

```
Z = (LRANDOM(I,12)-6)*S+X
```

ERROR CONDITIONS: If N is negative the program terminates with the message:

NEGATIVE ARGUMENT IN RANDOM.

## MOD

DEFINITION: This longreal function gives the value of the modulus (i.e. the absolute value) of the quantity specified on entry.

SPEC: Not required.

CALL: MOD(X)

| | |
|---|---|
| X | is a real or longreal expression. |

ERROR CONDITIONS: None.

## MULT MATRIX

DEFINITION: This routine forms the matrix product A = BC where B is NxP, C is PxM, and A, the product is NxM. The arrays used must be distinct.

SPEC: <u>externalroutinespec</u> MULT MATRIX(<u>longrealarrayname</u> A,B,C, <u>integer</u> N,P,M)

CALL: MULT MATRIX (A,B,C,N,P,M)

| | |
|---|---|
| A,B,C | are the names of three longreal two-dimensional arrays. Arrays B and C contain the rectangular matrices to be multiplied together. Matrix multiplication is performed on the matrices stored in arrays B and C. The resulting matrix is stored in array A. Arrays B and C are unchanged. |
| N,P,M | are integer expressions whose values give the upper bounds of the dimensions for the given arrays, the lower limits being taken to be 1. |

ERROR CONDITIONS: If the upper bounds of any of the arrays are zero or negative, the program terminates with the message:

```
MATRIX BOUND ZERO OR NEGATIVE
```

## MULT TR MATRIX

DEFINITION: This routine forms the transpose matrix product $A = BC^T$ where $C^T$ is the transpose of a given MxP matrix C. If matrix B is NxP, then A is NxM. The arrays used must be distinct.

SPEC: externalroutinespec MULT TR MATRIX (longrealarrayname A,B,C, integer N,P,M)

CALL: MULT TR MATRIX (A,B,C,N,P,M)

A,B,C are the names of three longreal two-dimensional arrays. The array C contains the matrix to be transposed, and array B contains the matrix by which the transpose is to be multiplied.
The matrix in array B is multiplied by the transpose of the matrix in C. The resulting matrix is stored in array A. Arrays B and C are unchanged.

N,P,M are the integer expressions whose values give the upper limits of the dimensions of the given arrays, the lower limits being taken to be 1. Note that C has dimensions M and P in that order.

ERROR CONDITIONS: If any of the upper bounds of the matrices is zero or negative, the program terminates with the message:

```
MATRIX BOUND ZERO OR NEGATIVE
```

## NEWLINE

DEFINITION: This routine produces a 'newline' character on the currently selected output stream.

SPEC: Not required.

CALL: NEWLINE

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## NEWLINES

DEFINITION: This routine produces the specified number of 'newline' characters on the currently selected output stream.

SPEC: Not required.

CALL: NEWLINES(N)

N is an integer expression, the last 8 bits of which indicate the number of 'newline' characters to be output.

n 'newline' characters are produced on the output stream. Note that n <= 0 has no effect.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## NEWPAGE

DEFINITION: This routine causes the 'form-feed' character to be put into the output stream.

SPEC: Not required.

CALL: NEWPAGE

No parameters.

The 'form-feed' character is put into the output stream. If the currently selected output stream goes to the Line Printer, then the paper is advanced to the head of a new page. The effect on an interactive terminal depends on the characteristics of the terminal.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## NEXT CH

DEFINITION: This integer function gives the ISO numerical value of the next character appearing on the currently selected input stream.

SPEC: Not required

CALL: NEXT CH

No parameters.

The ISO numerical value of the next character belonging to the full Edinburgh Standard Character Set appearing on the input stream is returned. A subsequent call of READ CH or NEXT CH will obtain the same character.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1

## NEXT ITEM

DEFINITION: This string function reads the next symbol from the currently selected input stream, as a one-character-string. The symbol may be read again, for example by NEXT ITEM or NEXT SYMBOL.

SPEC: Not required.

CALL: NEXT ITEM

No parameters.

The ISO-numeric value of the next symbol from the input stream is obtained. This symbol can be obtained again by either a 'READ ITEM' or a 'NEXT ITEM' instruction.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1.

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this routine the program terminates with the message:

SUBSTITUTE CHARACTER IN DATA

This error is trappable as event 3/1.

## NEXT SYMBOL

DEFINITION: This integer function gives the ISO numerical value of the next symbol appearing on the currently selected input stream.

SPEC: Not required.

CALL: NEXT SYMBOL

No parameters.

The ISO numerical value of the next symbol appearing on the input stream is returned. A subsequent call of READ SYMBOL or NEXT SYMBOL will obtain the same symbol.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1.

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this routine the program terminates with the message:

SUBSTITUTE CHARACTER IN DATA

This error is trappable as event 3/1.

## NL

DEFINITION: This integer function returns as a result the value of the internal code of the newline character.

SPEC: Not required.

CALL: NL

No parameters.

ERROR CONDITIONS: None.

## NULL

DEFINITION: An NxM null matrix is set up by this routine in a specified array.

SPEC: externalroutinespec NULL(longrealarrayname A, integer N,M).

CALL: NULL(A,N,M)

| | |
|---|---|
| A | is the name of a longreal two-dimensional array. All elements of the array are set to zero. |
| N,M | are the integer expressions whose values, n and m, give upper bounds for the dimensions of the given array, the lower bounds being taken to be 1. |

ERROR CONDITIONS: If either of the matrix bounds of the array is found to be zero or negative, the program terminates with the message:

MATRIX BOUND ZERO OR NEGATIVE

## OPENDA

DEFINITION: This routine opens the direct access file on logical channel (n).

SPEC: externalroutinespec OPENDA (integer N)

CALL: OPENDA(N)

| | |
|---|---|
| N | is an integer expression which gives the channel number and must be in the range allowed by the operating system being used. |

The file is made ready for reading or writing.

ERROR CONDITIONS: If a file is opened twice without being closed, the program terminates with the message:

FILE ALREADY OPEN

If an attempt is made to open a file with an invalid channel number or a channel which has not been defined, the program terminates with an appropriate error message.

Although the logical channel number must lie in the range appropriate to the system being used it must not conflict with any channel numbers assigned to either stream I/O or to sequential files.

The user must declare his direct access file requirements to the operating system as specified in the appropriate User's Guide.

## OPENSQ

DEFINITION: This routine opens a sequential file on the logical channel indicated.

SPEC: externalroutinespec OPENSQ (integer M)

CALL: OPENSQ(M)

M is an integer expression which specifies the channel number and must be in the range allowed by the operating system being used.

The file is made ready for reading or writing, but note that both options may not be exercised.

ERROR CONDITIONS: If a file is opened twice without being closed, the program is terminated with the message:

FILE ALREADY OPEN

If an attempt is made to open a file with an invalid channel number or a channel which has not been defined, the program terminates with an appropriate error message.

Although the logical channel number must lie in the range appropriate to the system being used it must not conflict with any channel numbers assigned to either stream I/O or direct access files.

The user must declare his sequential file requirements to the operating system as specified in the appropriate User's Guide.

## OUTPOS

DEFINITION: This integer function returns the position on the current line of the last character output. Immediately after a call of NEWLINE, OUTPOS will return the value zero.

SPEC: externalintegerfnspec OUTPOS

CALL: OUTPOS

No parameters.

ERROR CONDITIONS: None

## OUT STREAM

DEFINITION: This integer function returns the number of the stream currently selected for output.

SPEC: externalintegerfnspec OUTSTREAM

CALL: OUTSTREAM

No parameters.

ERROR CONDITIONS: None

## PI

DEFINITION: This longreal function returns the value of $\pi$ (3.141592653589793).

SPEC Not required.

CALL PI

No parameters.

ERROR CONDITIONS: None.

## PRINT

DEFINITION: This routine prints the value of the specified real number on the currently selected output medium in fixed-point form.

SPEC: Not required.

CALL: PRINT(X,M,N)

| | |
|---|---|
| X | is a longreal expression which is to be output. |
| M | is an integer expression indicating how many digits have to be output before the decimal point. |
| N | is an integer expression indicating how many digits have to be output after the decimal point. |

x is printed via the current output stream in fixed-point form, with m digits before, and n digits after the decimal point. Insignificant leading zeros are replaced by spaces and the sign is right justified.
Positive sign is represented by a space.
If more than m significant figures occur before the decimal point, the point will be displaced to the right and extra digits inserted. The total number of positions used in printing x is (1+m+1+n).

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## PRINT CH

DEFINITION: This routine places the n least significant bits of the parameter as the next character in the output stream: n is seven under VME/B and eight on EMAS.

SPEC: Not required.

CALL: PRINT CH (N)

N is an integer expression giving the ISO code representation of the character to be printed.

The character, represented in ISO code by the last n bits of N, is printed out on the currently selected output stream.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## PRINT FL

DEFINITION: This routine prints on the output medium the longreal value specified by the first parameter in floating-point form.

SPEC: Not required.

CALL: PRINT FL (X,N)

| | |
|---|---|
| X | is a longreal expression which is to be printed out. |
| N | is an integer expression indicating the number of digits to be printed after the decimal point. |

A floating-point real number is printed out using n+7 printing positions, with n digits after the decimal point. The number is standardised in the range 1 <= x < 10.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## PRINT STRING

DEFINITION: This routine outputs a string to the currently selected output stream.

SPEC: Not required.

CALL: PRINT STRING(S)

S is the string expression to be output.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## PRINT SYMBOL

DEFINITION: This routine prints on the currently selected output stream the symbol whose ISO code value is specified on entry.

SPEC: Not required.

CALL: PRINT SYMBOL(N)

N is an integer expression giving the ISO code representation of the symbol to be printed.

The symbol, represented in ISO code by the last 7 bits of N, is printed out on the currently selected output stream. If the symbol is not in the IMP extended character set, SUB is placed in the output stream.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## RADIUS

DEFINITION: This longreal function gives the value of the radius of a circle whose equation is of the form $X^2+Y^2= R^2$ (where R is the radius).

SPEC: Not required.

CALL: RADIUS(X,Y)

X,Y are real or longreal expressions.

The real or longreal value of the radius, i.e. the value of the positive square root of $(x^2+y^2)$, is returned.

ERROR CONDITIONS: If $x^2$ or $x^2+y^2$ is greater than the largest permissible real number an overflow condition will occur and the program will terminate with the message:

```
RADIUS ARGS TOO LARGE
```

## RANDOM

DEFINITION: This real function produces a set of random numbers either in a rectangular or Gaussian distribution.

SPEC: <u>externalrealfnspec</u> RANDOM(<u>integername</u> I, <u>integer</u> N)

CALL: RANDOM(I,N)

| | |
|---|---|
| I | is the name of an integer variable which must be initialised to any odd integer. Note that a value of I < 50001 may produce a sequence which is similar in the first few terms. |
| N | is an integer expression whose value controls the type of distribution: N = 1 gives a rectangular distribution, n > 1 gives a Gaussian distribution with a mean value of n/2 and a standard deviation of $\sqrt{\frac{n}{12}}$. For a Gaussian distribution a value of n greater than 10 is recommended. |

A real number in the range 0-n is returned. Note that I contains an integer random sequence in the range 0-($2^{31}$-1).

Users may wish to scale the distribution. To produce Z with Gaussian distribution, mean X, and standard deviation S, call RANDOM with N = 12 and scale thus:-

```
Z = (RANDOM(I,12)-6)*S+X
```

ERROR CONDITIONS: If N is negative the program terminates with the message:

```
NEGATIVE ARGUMENT IN RANDOM.
```

Wrong results occur if RANDOM instead of LRANDOM is called when <u>realslong</u> is in operation.

## READ

DEFINITION: This routine reads numerical data from the currently selected input stream and uses 'READ SYMBOL' (q.v.). The number read may be in fixed or floating point form.

SPEC: Not required.

CALL: READ (X)

X is the name of an integer or real variable.

The next number is taken from the input stream and stored in X. Numbers whose modulus is less than $2.4\times10^{-78}$ are treated as zero. Numbers whose modulus is greater than $7.2\times10^{75}$ will overflow.

ERROR CONDITIONS: If any characters other than digits + - @ . occur, the program terminates with the message:

```
SYMBOL IN DATA
```

followed by the invalid symbol. This error is trappable as event 4/1. In the current implementation, if this error is trapped the offending symbol may be obtained using READ SYMBOL.

If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

```
INPUT ENDED
```

This error is trappable as event 9/1.

## READ CH

DEFINITION: This routine reads a character from the input stream in internally coded form before any line reconstruction is done.

SPEC: Not required.

CALL: READ CH (N)

N is the name of an integer variable.

The next character on the input stream, belonging to the full Edinburgh Standard Character Set, is read in and stored in the location specified by N. The character is stored (in internal code) in the 8 least significant bits of N.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

```
INPUT ENDED
```

This error is trappable as event 9/1.

## READDA

DEFINITION: This routine reads data from a direct access file, connected as the channel specified, into the specified area, starting from the specified record on the file.

SPEC: <u>externalroutinespec</u> READDA (<u>integer</u> N, <u>integername</u> SECT, <u>name</u> BEGIN, END)

CALL: READDA (N,SECT,BEGIN,END)

| | |
|---|---|
| N | is an integer expression which specifies the channel number. |
| SECT | is an <u>integername</u> parameter whose value on entry specifies the record at which reading is to start, and on exit contains the number of the last record read from. |
| BEGIN<br>END | are variable names or array elements which specify the area into which data is to be read. |

Information is read from the direct access file on channel n starting at record sect and is written into the area starting at BEGIN and finishing at the end of END. This area is normally an array.

ERROR CONDITIONS: If ADDR(END) < ADDR(BEGIN), the program terminates with the message:

ADDRESSES INSIDE OUT.

If an attempt is made to access a record on the file which does not exist, the program terminates with the message:

RECORD NUMBER OUT OF RANGE

If the specified channel is not open, the program terminates with:

FILE NOT OPEN

If an invalid record size has been specified in a file definition, the program terminates with the message:

INVALID RECORD SIZE

If an input or output request is not consistent with the file definition, the program terminates with the message:

INVALID OPERATION ON FILE

If an attempt is made to read from a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

The record size for direct access files is fixed at 1024 bytes and, on creation, the complete file is filled with the unassigned pattern.

## READ ITEM

DEFINITION: This routine reads the next symbol from the currently selected input stream, and sets this up as a one-character string.

SPEC: Not required.

CALL: READ ITEM (S)

| | |
|---|---|
| S | is the name (of type string) of the location in which the symbol is stored. |

The ISO-numeric value of the next symbol from the input stream is stored in S.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1.

Invalid characters in the input stream under VME/B only are replaced by SUB.

## READLSQ

DEFINITION: This routine reads information from a sequential file, connected as the channel specified on entry, into the array specified on entry.

SPEC: externalroutinespec READLSQ(integer N, name BEGIN,END, integername L)

CALL: READLSQ(N,BEGIN,END,L)

| | |
|---|---|
| N | is an integer expression specifying the channel number and whose value must be in the range allowed by the operating system being used. |
| BEGIN<br>END | are variable or array element names which specify the area into which the information is to be read. |
| L | is the name of an integer which will return the length in bytes of the record just read. If an attempt is made to read more data than has been provided on the input stream L will be set to the value zero. |

ERROR CONDITIONS: If the specified channel is not open the program will terminate with the message:

FILE NOT OPEN

If ADDR(END) < ADDR(BEGIN), the program will terminate with the message:

ADDRESSES INSIDE OUT

If an attempt is made to read from a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

## READSQ

DEFINITION: This routine reads information from a sequential file, connected as the channel specified on entry, into the array specified on entry.

SPEC: <u>externalroutinespec</u> READSQ (<u>integer</u> M, <u>name</u> BEGIN,END)

CALL: READSQ(M,BEGIN,END)

| | |
|---|---|
| M | is an integer expression which specifies the channel number and whose value must be in the range allowed by the operating system being used. |
| BEGIN } END } | are variable names or array elements which specify the area into which the information is to be read. |

Information is read from the channel specified on entry into the area starting at location BEGIN, and finishing at the end of location END. The area specified is normally an array.

ERROR CONDITIONS: If the specified channel is not open the program will terminate with the message:

FILE NOT OPEN

If ADDR(END) < ADDR(BEGIN), the program will terminate with the message:

ADDRESSES INSIDE OUT

If an attempt is made to read beyond the end of a file, the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1.

If an attempt is made to read from a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

## READ STRING

DEFINITION: This routine reads the next string from the currently selected input stream.

SPEC: Not required.

CALL: READ STRING (S)

S is the name (of type string) of the location in which the string of symbols is stored.

The string of symbols (which must be enclosed within double quotes and must contain 255 or less symbols) is read into S.

ERROR CONDITIONS: If the first significant symbol read by READ STRING (ignoring spaces and newlines) is not a double quote, the program terminates with the message:

```
SYMBOL INSTEAD OF STRING
```

If the declared maximum length of the stringname parameter is insufficient to hold the string, the program terminates with the message:

```
CAPACITY EXCEEDED
```

If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

```
INPUT ENDED
```

This error is trappable as event 9/1.

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this routine the program terminates with the message:

```
SUBSTITUTE CHARACTER IN DATA
```

This error is trappable as event 3/1.

## READ SYMBOL

DEFINITION: This routine reads the next symbol which appears on the currently selected input stream.

SPEC: Not required.

CALL: READ SYMBOL(N)

N is the name of an integer or byteinteger variable.

The next symbol on the input stream is read in and stored in the location specified by N.
The symbol is stored (in internal code) in the 7 least significant bits of N.
All non-printing characters, (including CR), are ignored.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

```
INPUT ENDED
```

This error is trappable as event 9/1.

Invalid characters in the input stream are passed through by EMAS, but under VME/B are replaced by SUB and if SUB is detected by this routine the program terminates with the message:

```
SUBSTITUTE CHARACTER IN DATA
```

This error is trappable as event 3/1.

## REAL

DEFINITION: This map enables the user to access a particular 4-byte location whose address is specified by the parameter.

SPEC: Not required.

CALL: REAL(N)

N is an integer expression giving the address of the location to be accessed.

This may be used to access a correctly aligned 4-byte location, as though it held a real variable, or to store a real value at the specified address:

```
integer L
real R
R = 54.3
REAL(ADDR(L)) = R;          !L now contains 54.3 held as a floating
                            !point number
PRINT(L,6,1);               !The bit pattern in L is treated as that
                            !of an integer value.  54.3 will not be
                            !the value printed.
PRINT(REAL(ADDR(L)),6,1);   !54.3 is printed.
```

Note that REAL is always single length and is not affected by realslong

ERROR CONDITIONS: If the specified location is not correctly full-word aligned, address truncation occurs when the routine attempts to access it.

## RECORD

DEFINITION: This mapping function renders the absolute address specified on entry in a form such that it may be assigned to a recordname variable using the '==' assignment operator.

SPEC: Not required.

CALL: RECORD(ADR)

ADR is an integer expression giving the absolute address of the core area to be treated as a record.

The address, in a form such that it can be assigned to a recordname variable using the '==' assignment operator, is returned.

The following example illustrates the use of a record map:

```
integerarray NAME(1:11)
recordformat R1(integer I,J, byteinteger K,L,M,N, string (15) S,T)
recordname R(R1)
R == RECORD (ADDR(NAME(1)))

R_I is a reference to NAME(1)
R_K is a reference to the top byte of NAME(3)
```

For more information on records see the Edinburgh IMP Language Manual.

ERROR CONDITIONS: If the address specified is not double word aligned the effects are undefined.

## SELECT INPUT

DEFINITION: This routine causes subsequent input to be taken from the stream specified on entry.

SPEC: Not required.

CALL: SELECT INPUT(N)

N is an integer expression giving the number of the input stream to be selected, and whose value must lie in the range allowed by the operating system being used.

The current input is connected to the stream specified by N. All further input is obtained via the new stream.
Under VME/B any unused information from the current record is lost during execution of this routine.

ERROR CONDITIONS: If no stream is defined on entry to this routine the program terminates with the message:

STREAM NOT DEFINED

If the stream defined has previously been defined for output, the program terminates with the message:

STREAM IN USE

## SELECT OUTPUT

DEFINITION: This routine causes subsequent output to be sent to the stream specified on entry.

SPEC: Not required.

CALL: SELECT OUTPUT(N)

N is an integer expression giving the number of the output stream selected and must be in the range allowed by the operating system being used.

Under VME/B the current record is output to the previously selected stream unless the last symbol output before the call of this routine was a 'newline' symbol. In the latter case, nothing is output. All subsequent records are sent to the new stream.

ERROR CONDITIONS: If no stream is defined on entry to this routine the program terminates with the message:

```
STREAM NOT DEFINED
```

If the stream defined has been previously defined for input, the program terminates with the message:

```
STREAM IN USE
```

If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## SET MARGINS

DEFINITION: This routine allows the margins of either the current input or the current output user-defined stream (i.e. any stream lying between 1 and 98 inclusive) to be changed. This routine is only available under VME/B.

SPEC: Not required.

CALL: SET MARGINS (M,LEFT,RIGHT)

| | |
|---|---|
| M | is an integer expression which gives the number of the stream whose margins are to be changed. |
| LEFT | is an integer expression specifying the left hand margin and can take values 1 <= left <= right. |
| RIGHT | is an integer expression specifying the right hand margin and can take values left <= right <= 160 for input streams or left <= right <= 132 for output streams. |

If SET MARGINS operates on the current input stream, the default values for which are left = 1, right = 80, (with an exception for Stream 99: right = 132), left and right may take values which satisfy 1 <= left <= right <= 160. The effect of changing input margins is truncation of the input record at the beginning if left is increased or truncation of the input record at the end if right is reduced. If SET MARGINS operates on the current output stream, the default values for which are left = 1, right = 120, left and right may take values which satisfy 1 <= left <= right <= 132. The effect of altering the left hand margin is to inset the output record, and the effect of reducing the right hand margin is to output any overflow on the next line. The margins, once set on a given stream, remain as characteristics of that stream until the next call of SET MARGINS for the stream.

ERROR CONDITIONS: If m is not the current input or output stream; if m < 1 or m > 98; the program terminates with the message:

```
STREAM NOT DEFINED
```

If left > right or if either lies outside their defined limits, the program terminates with the message:

```
INVALID MARGINS
```

## SHIFTC

DEFINITION: This integer function cyclically shifts a given bit pattern by a specified number of bits.

SPEC: externalintegerfnspec SHIFTC (integer N,M)

CALL: SHIFTC (N,M)

| | |
|---|---|
| N | is an integer expression whose binary pattern is to be shifted. |
| M | is an integer expression indicating the number of places n is to be shifted. Note that m may be positive, indicating a shift to the left, or negative, indicating a shift to the right. |

The bit pattern of the integer in location N is shifted m places, left or right according to the sign (+ or - respectively) of m.

Bits lost off one end of the word reappear in the same order at the other end of the word.

ERROR CONDITIONS: If the absolute value of m is greater than 32, the program terminates with the message:

ILLEGAL SHIFT

## SHORTENI

DEFINITION: This integer function changes an operand in an expression from type longinteger to integer.

SPEC: Not required.

CALL: SHORTENI(L)

| | |
|---|---|
| L | is a longinteger expression. |

ERROR CONDITIONS: In checking mode overflow will occur if L is too large to be contained in an integer.

## SHORTENR

DEFINITION: This longreal function changes an operand in an expression from type longlongreal to longreal.

SPEC: Not required.

CALL: SHORTEN(R)

| | |
|---|---|
| R | is a longlongreal expression. |

ERROR CONDITIONS: None.

## SIN

DEFINITION: This longreal function gives the value of the sine of the quantity specified on entry. This quantity must be in radians.

SPEC: Not required.

CALL: SIN(X)

X is a real or longreal expression.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than $3.53 \times 10^{15}$, the result would be wholly inaccurate and the program terminates with the following message:

SIN ARG OUT OF RANGE

## SKIP SYMBOL

DEFINITION: This routine passes over the next symbol which appears on the currently selected input stream, without reading it.

SPEC: Not required.

CALL: SKIP SYMBOL

No parameters.

ERROR CONDITIONS: If an attempt is made to read more data than has been provided on the input stream, the program terminates with the message:

INPUT ENDED

This error is trappable as event 9/1.

Invalid characters in the input stream under VME/B only are replaced by SUB.

## SOLVE LN EQ

DEFINITION: This routine solves a linear system of equations Ax = b for vector x. A, an NxN matrix, and b, a vector, are destroyed. The method of Gaussian elimination with partial pivoting is used.

SPEC: externalroutinespec SOLVE LN EQ (longrealarrayname A,B, integer N, longrealname DET)

CALL: SOLVE LN EQ (A,B,N,DET)

| | |
|---|---|
| A | is the name of a longreal two-dimensional array which contains square matrix A on input. |
| B | is the name of a longreal one-dimensional array which contains vector b on input. |
| N | is the integer expression whose value, n, gives an upper limit for the dimensions of A and B, the lower limits being taken to be 1. |
| DET | is the name of a longreal variable which is set by the routine to the value of the determinant of the matrix in array A. |

The solution of the equations, x, is placed in B, and matrix a is destroyed.
The value of the determinant of a is placed in DET.
Note that large errors may occur if the matrix is ill-conditioned and, if this is suspected, perturbed matrices should be examined.

ERROR CONDITIONS: If n is less than or equal to zero, the program terminates with the message:

```
SOLVE LN EQ DATA FAULT N = n
```

If the matrix is not invertible, the value of DET will be zero on exit.

## SPACE

DEFINITION: This routine produces one 'space' character on the currently selected output stream.

SPEC: Not required.

CALL: SPACE

No parameters.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## SPACES

DEFINITION: This routine produces the specified number of 'space' characters on the currently selected output stream.

SPEC: Not required.

CALL: SPACES(N)

N is an integer expression the last 8 bits of which indicate the number of 'space' characters to be produced.

n 'space' characters are produced on the output stream.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## SQRT

DEFINITION: This longreal function gives the value of the positive square root of the quantity specified on entry.

SPEC: Not required.

CALL: SQRT(X)

X is a real or longreal expression.

The real or longreal value of the positive square root of the quantity specified by X is returned.

ERROR CONDITIONS: If a negative argument is passed to this routine, the program terminates with the message:

```
SQRT ARG NEGATIVE
```

## STRING

DEFINITION: This map enables the user to access, as a string, a location whose address is specified by the parameter.

SPEC: Not required.

CALL: STRING(N)

N is an integer expression giving the absolute address in core of the required location.

This map may be used to access the contents of a location as though it held a string variable, or to store a string at the specified address:

Note that the first byte of a string is taken as the length of the string.

```
string(10) S
byteintegerarray A(0:10)
S = "ABCDEFGHIJ"
STRING(ADDR(A(0))) = S;     !now A(0) = 10,A(1) = 'A'
                            !A(2) = 'B'--- A(10) = 'J'
A(0) = 4
S = STRING(ADDR(A(0)));     !now S = "ABCD"
```

ERROR CONDITIONS: None.

## SUB MATRIX

DEFINITION: This routine subtracts one NxM matrix from another.

SPEC: externalroutinespec SUB MATRIX (longrealarrayname A,B,C, integer N,M)

CALL: SUB MATRIX(A,B,C,N,M)

| | |
|---|---|
| A,B,C | are the names of three longreal two-dimensional arrays. The matrix in array C is subtracted from the matrix in B. Array A contains the matrix difference of matrices in arrays B and C. Arrays B and C are unchanged. |
| N,M | are the integer expressions whose values give the upper limits for the dimensions of A, B and C, the lower limits being taken to be 1. |

ERROR CONDITIONS: If the array bounds of any of the matrices are zero or negative, the program terminates with the message:

MATRIX BOUND ZERO OR NEGATIVE

## TAN

DEFINITION: This longreal function gives the value of the tangent of the quantity specified on entry. This quantity must be in radians.

SPEC: Not required.

CALL: TAN(X)

X is a real or longreal expression.

The real or longreal value of the tangent of x is returned.

ERROR CONDITIONS: If the modulus of the argument of this function is greater than 3.53 x $10^{15}$ the result would be wholly inaccurate, and the program terminates with the message:

```
TAN ARG OUT OF RANGE
```

## TIME

DEFINITION: This string function yields the current time of day.

SPEC: externalstringfnspec TIME

CALL: TIME

No parameters.

The time is returned in the form hh:mm:ss, and may be printed out using PRINTSTRING.

ERROR CONDITIONS: None.

## TO STRING

DEFINITION: This string function creates a one-character string.

SPEC: Not required.

CALL: TO STRING(N)

N the least significant 7 bits of the integer expression N represent the ISO code value of the character to be placed in the string.

A one-character string, containing the character specified by N, is created.

ERROR CONDITIONS: None.

## TRANS MATRIX

DEFINITION: This routine sets up, in a specified two-dimensional array, the NxM transpose of a given MxN matrix. The specified arrays must be distinct.

SPEC: externalroutinespec TRANS MATRIX (longrealarrayname A,B, integer N,M)

CALL: TRANS MATRIX (A,B,N,M)

| | |
|---|---|
| A,B | are the names of two longreal two-dimensional arrays. Array B contains the rectangular matrix to be transposed. The transpose of the matrix stored in array B is set up in the array A. Array b is unchanged. |
| N,M | are the integer expressions whose values, n and m, give upper bounds for the dimensions of the specified arrays, the lower bounds being taken to be 1. |

ERROR CONDITIONS: If the upper bounds of either matrix are zero or negative, the program terminates with the message:

MATRIX BOUND ZERO OR NEGATIVE

## UNIT

DEFINITION: An NxN unit matrix is set up in a specified array.

SPEC: externalroutinespec UNIT (longrealarrayname A, integer N).

CALL: UNIT (A,N)

| | |
|---|---|
| A | is the name of a two-dimensional longreal array. The diagonal elements of A are set to 1 and all other elements to zero. |
| N | is the integer expression whose value, n, gives an upper bound for the dimensions of A, the lower bounds being taken to be 1. |

ERROR CONDITIONS: If n is zero or negative, the program terminates with the message:

MATRIX BOUND ZERO OR NEGATIVE

## WRITE

DEFINITION: This routine prints on the output medium the integer value specified by the first parameter using one character position for the sign and the next n positions for the digits (where n represents a number specified on entry to the routine).

SPEC: Not required.

CALL: WRITE(X,N)

| | |
|---|---|
| X | is an integer expression whose value, x, is to be printed out. |
| N | is an integer expression specifying the number of digits to be printed. |

An integer is printed out with n significant figures preceded by a sign.
Non-significant leading zeros are suppressed, being replaced by spaces, and a positive sign is indicated by a space.
If x has more than n significant figures, or if n is zero or negative, all the figures are printed out, but the righthand end of the number is then out of alignment. The sign always immediately precedes the most significant digit.
x is accurate to 11 significant figures.

ERROR CONDITIONS: If an attempt is made to output more data than has been specified for the current output stream, the program terminates with the message:

```
OUTPUT EXCEEDED
```

## WRITEDA

DEFINITION: This routine writes the information held in the specified area onto the specified file, starting at the specified record.

SPEC: externalroutinespec WRITEDA (integer N, integername SECT, name BEGIN, END)

CALL: WRITEDA(N,SECT,BEGIN,END)

| | |
|---|---|
| N | is an integer expression which specifies the channel number and must be in the range allowed by the operating system being used. |
| SECT | is an integername parameter whose value on entry specifies the record at which writing is to start and, on exit, contains the number of the last record written. |
| BEGIN } END } | are variable names or array elements which specify the area from which data is to be written. |

The information contained in core, from BEGIN to END inclusive, is written to the direct access file on channel n starting at record sect. Writing always commences at the start of a record and if the information written does not fill a complete number of records then the remainder of the last record written is undefined.

ERROR CONDITIONS: If ADDR(END) < ADDR(BEGIN), the program terminates with the message:

ADDRESSES INSIDE OUT

If an attempt is made to access a non-existent record on the specified file, the program terminates with the message:

RECORD NUMBER OUT OF RANGE

If the specified channel is not open, the program terminates with:

FILE NOT OPEN

If an attempt is made to write to a file with an invalid channel number or one which has not been defined, the program terminates with an appropriate error message.

If an invalid record size has been specified in a file definition, the program terminates with the message:

INVALID RECORD SIZE

If an input or output request is not consistent with the file definition, the program terminates with the message:

INVALID OPERATION ON FILE

## WRITESQ

DEFINITION: This routine writes the information held in the area specified on entry onto the next logical record of the sequential file on the channel specified on entry.

SPEC: <u>externalroutinespec</u> WRITESQ (<u>integer</u> M, <u>name</u> BEGIN,END)

CALL: WRITESQ(M,BEGIN,END)

| | |
|---|---|
| M | is an integer expression which specifies the channel number and must be in the range allowed by the operating system being used. |
| BEGIN<br>END | are variable names or array elements which specify the area from which the information is to be taken. |

The information contained in core, between BEGIN and END inclusive, is written in a new logical record on to the file specified by channel M. The area specified is normally an array.

ERROR CONDITIONS: If the specified channel is not open the program terminates with the message:

```
FILE NOT OPEN
```

If ADDR(END) < ADDR(BEGIN), the program terminates with the message:

```
ADDRESSES INSIDE OUT
```

# 4

# Cross calling between ALGOL, FORTRAN and IMP

# MIXED LANGUAGE PROGRAMMING

## 4.1 Introduction

One of the design aims of the Edinburgh IMP, FORTRAN and ALGOL implementations has been to allow programs to include routines or procedures from all three languages. To this end routines in these languages use, where possible, the same conventions for passing parameters. This section first discusses the underlying principles governing parameter passing (with examples) and then systematically defines the means of communicating between routines written in different languages.

## 4.2 Actual-formal parameter correspondence

By a formal parameter (sometimes referred to as a dummy argument) we mean an identifier in a procedure which is replaced by another identifier or an expression when the procedure is called. In a FORTRAN subroutine beginning with

```
SUBROUTINE FRED(A,B)
```

A and B are the formal parameters. An equivalent ALGOL procedure begins with

```
%PROCEDURE FRED(A,B);
```

If the above procedure is called by FRED(X,Y*Z) the formal parameters A and B are connected in some way to the actual parameters X and Y*Z.

When a procedure in any language is called, a parameter list is passed to it. The procedure uses this list to make the actual-formal parameter correspondence. As well as the actual parameters there may be implicit parameters which the programmer is not aware of. One of these is the return address.

Depending on the language and the type of parameter the list of parameters may contain:
- a) the actual parameter
- b) the address of a variable holding the parameter
- c) a descriptor (or dope-vector) describing the parameter
- d) the address of code to evaluate the parameter address.

### 4.2.1 Call by Reference

The parameter in the parameter list is an address used to access (or reference) the location containing the actual parameter (the temporary location holding its value in the case of an expression).

Examples are: arrays in FORTRAN
<u>name</u> type parameters in IMP

### 4.2.2 Call by Value

The actual parameter is evaluated by the calling procedure and placed directly in the parameter list and this value is used by the called procedure. Thus there is no way for the called procedure to change the value of the actual parameter in the calling procedure. Examples are: <u>real</u> and <u>integer</u> type parameters in IMP.

### 4.2.3 Call by Value Result

The address of the actual parameter is placed in the parameter list. On entry to the called procedure a local variable is created which is initialised to the actual value. All computation which refers to the corresponding formal parameter then takes place using the local variable. On exit from the called procedure the value of the local variable resulting from the execution of the procedure is used to update the actual parameter.

This technique is used in FORTRAN, although the user can specify that a particular formal parameter should be treated as a call by Reference. In terms of efficiency, treatment of formal parameters as local variables allows the benefit of direct, rather than indirect, accessing although this benefit is offset by the overheads of the copy-in and copy-back sequences.

## 4.2.4 Call by Name

The Algol report implies that the use of a parameter called by name is equivalent to the textual substitution of the actual parameter for each appearance of the formal parameter in the procedure body. It cannot be implemented efficiently in this way. The usual method of implementation is to produce, for each actual parameter which replaces a formal name parameter, a section of code which evaluates the actual parameter and returns the address where the resultant value is stored. Such sections of code are called thunks. It is the address of the thunk which is placed in the parameter list by the calling procedure. In producing code for the procedure itself each reference to a name parameter will be handled by making a call of the corresponding thunk via its address in the parameter list, followed by a reference to the value using the address returned by the thunk.

## 4.2.5 Array Name as Actual Parameter

In this case both actual and formal parameters must be arrays to make sense. The address of the first array element (for languages like FORTRAN which do not require a dope vector) or the address of the dope vector is placed in the parameter list by the calling procedure, and this is used by the called procedure to reference array elements.

Note on dope vectors: In FORTRAN the upper and lower bounds of arrays are known at compile time; thus the compiler can allocate storage for arrays and generate code to access them at compile time. In Algol and IMP, subscript bounds can be variable - hence storage cannot be allocated until run-time. Thus at compile time the compiler allocates a fixed amount of space in current data area for a template or dope vector (fixed in size) and will produce code to carry out array accesses via this dope vector. Code will also be inserted by the compiler to evaluate the subscript bounds, allocate the required number of locations, and fill the dope vector with appropriate information.

## 4.2.6 Procedure Name as Actual Parameter

The parameter contains sufficient information to enable the called procedure to enter the procedure passed as a parameter, locating its local data and any global context that may be applicable.

## 4.2.7 Summary

The table of Languages vs Parameter types below summarises the options available with reference to parameter passing.

| Parameter Types / Language | Call by Reference | Call by Name | Call by Value | Call by Value result |
|---|---|---|---|---|
| IMP | * |  | * |  |
| FORTRAN | * |  |  | * |
| ALGOL 60 |  | * | * |  |

Notes

Algol 60 uses call-by-value and call-by-name. In call-by-value the formal parameters of a procedure are treated as local variables which are initialised to the values of the actual parameter.

The difference between a call by reference and a call by name is the following: the address of an actual parameter called by reference is evaluated only once, before the procedure is actually called, while for a call by name the address is evaluated each time the formal parameter is referenced in the procedure body.

## 4.3 Calling IMP routines from FORTRAN programs and FORTRAN subprograms from IMP

Conventions for passing parameters between the two languages follow with examples and restrictions.

### 4.3.1 Routine and Function Calls

IMP routines which are to be called explicitly from a FORTRAN routine must be defined as %EXTERNAL, and similarly FORTRAN routines must be specified as %EXTERNAL within IMP routines from which they are called.

e.g. an IMP routine

```
%EXTERNALROUTINE SUB(...)
.
.
.
%END
```

could be called from a FORTRAN routine by

```
CALL SUB(...)
```

or e.g. a FORTRAN subroutine

```
SUBROUTINE SUBF(....)
.
.
.
END
```

could be called from IMP by

```
%EXTERNALROUTINESPEC SUBF(....)
SUBF(....)
.
.
```

Functions are treated similarly. The permitted IMP function types are %INTEGERFN, %REALFN, %LONGREALFN and %LONGLONGREALFN corresponding to FORTRAN INTEGER*4 FUNCTION, REAL*4 FUNCTION, REAL*8 FUNCTION and REAL*16 FUNCTION respectively.

### 4.3.2 Scalar Parameters

The permitted IMP parameter types are %INTEGERNAME, %LONGINTEGERNAME, %REALNAME, %LONGREALNAME and %LONGLONGREALNAME, corresponding to FORTRAN parameters INTEGER*4, INTEGER*8, REAL*4, REAL*8 and REAL*16.

e.g. the FORTRAN routine defined by

```
SUBROUTINE SUB1 (X,Y,I)
REAL*8 X,Y
.
.
.
END
```

could be accessed from an IMP routine containing the specification

```
%EXTERNALROUTINESPEC SUB1 (%LONGREALNAME X,Y,%INTEGERNAME I)
```

### 4.3.3 Array Parameters

FORTRAN and IMP conventionally set up array parameters in very different ways. FORTRAN passes the address of the first element of an array and defines the structure within the called routine while IMP passes a descriptor defining the location and structure of the array.

The following techniques are recommended for handling array parameters.

(a) IMP calling FORTRAN

In the routine specification instead of specifying an %ARRAYNAME parameter define a %NAME parameter of the appropriate type and set as actual parameter the first element of the array.

e.g. Where the called routine is

```
      SUBROUTINE R(A,...)
      DIMENSION A(50)
      .
      .
      .
      END
```

the calling routine should include statements of the form

```
      %EXTERNALROUTINESPEC R(%REALNAME B...)
      %REALARRAY B(1:50)
      .
      .
      .
      R (B(1),...)
```

References to elements of the array A in FORTRAN routine R will access the array B in the IMP routine.

(b) FORTRAN calling IMP

Passing array parameters FORTRAN passes only the address of the first element hence a called IMP routine has to construct its own description. This may be done as follows:

FORTRAN calling routine:

```
      REAL Y(50,10)
      .
      .
      .
      CALL SUB 2(Y,...)
```

IMP called routine:

```
      %EXTERNALROUTINE SUB 2(%REALNAME AX,...)
      %REALARRAYNAME X
      %REALARRAYFORMAT P (1:50, 1:10)
      X==ARRAY(ADDR(AX),P)
```

References to elements of array X within the IMP routine will access the array Y defined in the FORTRAN routine. The FORTRAN call sets the address of Y(1,1), i.e. the first element, as the actual parameter.

By defining the corresponding IMP parameter as %REALNAME, ADDR(AX) indicates where this address is stored and which, with the array format specification P, is used by the special mapping function ARRAY to construct an array descriptor for X.

IMP follows the FORTRAN convention for the storage of multi-dimensional arrays, i.e. such that the first subscript varies the fastest.

### 4.3.4 Routine and Function Parameters

These may be freely passed between IMP and FORTRAN subject to the type restriction defined in 4.3.1. In the case of IMP a routine or function passed as a parameter may be either external or internal.

### 4.3.5 Restrictions

In the case of IMP calling FORTRAN there is no provision for passing

(i) any value type variables
(ii) strings or records.

Note however that byte integer variables passed by name become LOGICAL*1.

In the case of FORTRAN calling IMP, LOGICAL*4 may be passed if they are treated as %INTEGERNAME or LOGICAL*1 may be passed if they are treated as %BYTEINTEGERNAME in IMP.

4.3.6 The following examples should assist in clarifying the conventions discussed previously.

4.3.6.1 Example of FORTRAN calling IMP

The following main program in FORTRAN calls a separately compiled IMP routine.

```
      DOUBLE PRECISION A(10),B,ONE
      INTEGER R
      LOGICAL C
      DATA ONE/1.0D0/
      DO 16 I=1,10
   16 A(I)=ONE
      C=.TRUE.
      R=1
      B=0.5D0
      CALL IMPCAL(A,B,R,C)
      WRITE(7,18) A
   18 FORMAT(1X,10(F4.2,1X))
      WRITE(6,19) B,R,C
   19 FORMAT(' B:=',F4.2,2X,'R:=',I2,2X,'C:=',L4)
      STOP
      END
```

The IMP called routine is coded as below:

```
%EXTERNALROUTINE IMPCAL(%LONGREALNAME X,Y,%INTEGERNAME S,T)
%INTEGER I
%LONGREALARRAYNAME Z
%LONGREALARRAYFORMAT P(1:10)
Z==ARRAY(ADDR(X),P)
%IF T=1 %THEN ->L1
S=3;Y=1.5
%CYCLE I=1,1,10
Z(I)=0
%REPEAT
-> L2
L1:S=2;Y=2.5
%CYCLE I=1,1,10
Z(I)=2
%REPEAT
L2:%END
%ENDOFFILE
```

4.3.6.2 Results

The output from this program is:

```
2.00 2.00 2.00 2.00 2.00 2.00 2.00 2.00 2.00 2.00

B:=2.50  R:= 2  C:=   T
```

Notes

1. All the formal parameters of the routine are name type parameters.
2. Corresponding to the actual array parameter A in FORTRAN is the formal parameter X in IMP which is a %LONGREALNAME type parameter, and not a %LONGREALARRAYNAME type.
3. FORTRAN logical variables appear to IMP as integers taking the value of 0 for .FALSE. and 1 for .TRUE..

### 4.3.6.3 Example of IMP calling FORTRAN

The called FORTRAN subroutine FCAL(N,X,LOGW,FUNCT) has the following formal parameters

N - an INTEGER quantity

X - a double precision one-dimensional real array of at least N elements

LOGW - a one-dimensional logical array

FUNCT - a SUBROUTINE supplied by the user whose specification is

```
SUBROUTINE FUNCT(N,X)
DOUBLE PRECISION X(N)
```

The IMP main program which calls the independently compiled FORTRAN subroutine is as follows:

```
%REALSLONG
%BEGIN
    %INTEGER N,I
    %REALARRAY X(1:10)
    %INTEGERARRAY LOGW(1:10)
    %EXTERNALROUTINESPEC FCAL(%INTEGERNAME N,%LONGREALNAME X,%C
    %INTEGERNAME LOGW,%ROUTINE FUNCT)
          %ROUTINE FUNCT(%INTEGERNAME N,%LONGREALNAME X)
             %INTEGER I
             %LONGREALARRAYNAME PX
             %LONGREALARRAYFORMAT FPX(1:N)
             PX==ARRAY(ADDR(X),FPX)
             %CYCLE I=1,1,N
             PX(I)=2.5
             %REPEAT
          %END
    N=8
    %CYCLE I=1,1,10
       X(I)=1.5
       LOGW(I)=1
    %REPEAT
    FCAL(N,X(1),LOGW(1),FUNCT)
    %CYCLE I=1,1,10
    PRINT(X(I),2,2);SPACES(2)
    %REPEAT
    NEWLINES(2)
    %CYCLE I=1,1,10
    WRITE(LOGW(1),2);SPACES(2)
    %REPEAT
    %ENDOFPROGRAM
```

The called FORTRAN subroutine is as below:

```
   SUBROUTINE FCAL(N,X,LOGW,FUNCT)
   REAL*8 X(N)
   LOGICAL LOGW(N)
   IF(.NOT.LOGW(1)) GOTO 17
   CALL FUNCT(N,X)
   RETURN
17 DO 21 I=1,N
21 X(I)=3.5D0
   RETURN
   END
```

### 4.3.6.4 Results

The output from this program is:

```
2.50   2.50   2.50   2.50   2.50   2.50   2.50   2.50   1.50   1.50

1   1   1   1   1   1   1   1   1
```

Notes

1. The subroutine FUNCT is coded in IMP to illustrate a situation where an IMP program calls a FORTRAN subroutine, which in turn calls an IMP routine.

   Alternatively, of course, the subroutine FUNCT could have been written in FORTRAN as per its specification.

2. In the statement containing the IMP call to the FORTRAN subroutine, viz

   ```
   FCAL(N,X(1),LOGW(1),FUNCT)
   ```

   the addresses of the first elements of the arrays are passed to the called subroutine. Had X, for instance, been a two-dimensional array then the call would have been

   ```
   FCAL(N,X(1,1),LOGW(1),FUNCT)
   ```

   It is advisable, if only for clarity, to declare the lower bounds of the arrays as 1 in IMP. The dimension(s) are not known in the called routine but may be passed as further integer parameters.

## 4.4 Calling ALGOL from FORTRAN and FORTRAN from ALGOL

It is possible to call ALGOL procedures from FORTRAN. As for external subprograms any ALGOL procedure name used as an actual parameter in a CALL statement or function reference must be given in an EXTERNAL statement in the program unit in which the name is used as an actual parameter.

The EXTERNAL statement has the form:

EXTERNAL name$_1$, name$_2$, ....., name$_n$

where any or all of the names may be ALGOL procedure names that are used as actual parameters in the program unit containing the EXTERNAL statement.

If an array parameter is to be passed to an ALGOL procedure then the procedure name must appear in a special form of the EXTERNAL statement which has the form:

EXTERNAL/ALGOL/name$_1$, name$_2$, ....., name$_n$

and this statement must be the first statement in the FORTRAN program.

There are two types of ALGOL procedures which correspond to FORTRAN functions and subroutines. The appropriate method of referencing should be used depending on the type of ALGOL procedure. ALGOL function procedures return a value of a particular type; valid correspondence of ALGOL and FORTRAN types of values is given in Section 4.4.1. ALGOL subroutine type procedures are referenced in FORTRAN programs by use of the CALL statement.

In Sections 4.4.1.1 and 4.4.1.2 tables of the correspondence between FORTRAN and ALGOL data types is provided. Since it is possible from FORTRAN to reference an ALGOL procedure that in turn calls a FORTRAN subprogram information on correspondence between ALGOL and FORTRAN data types is provided in Sections 4.4.2.1 and 4.4.2.2. Note however that in the case of ALGOL calling FORTRAN, a dummy procedure heading must be provided for each subprogram that may be accessed, and furthermore, the heading must be followed by the pseudo basic symbol <u>fortran</u> instead of <u>algol</u>. The dummy procedure heading will include a list of ALGOL type specifications corresponding to the types of FORTRAN parameters to the subroutine (see examples in Section 4.4.2.3).

### 4.4.1 FORTRAN calling ALGOL

To call an ALGOL procedure from a FORTRAN program unit it is necessary to ensure that the parameters, and in the case of functions, function results correspond to suitable items in the program. In FORTRAN terms the correspondence must be in terms of type and actual length. Valid correspondences for parameters are given in Section 4.4.1.1 while correspondences for function results follow in Section 4.4.1.2.

4.4.1.1 Valid Parameter Correspondence

| FORTRAN | ALGOL |
|---|---|
| REAL*8 | real |
| INTEGER*4 | integer |
| LOGICAL*4 | Boolean |
| INTEGER*4 FUNCTION | integer procedure |
| REAL*8 FUNCTION | real procedure |
| LOGICAL*4 FUNCTION | Boolean procedure |
| REAL*8 ARRAY | array or real array |
| INTEGER*4 ARRAY | integer array |
| LOGICAL*4 ARRAY | Boolean array |
| SUBROUTINE | procedure |

4.4.1.2 Valid Function Result Correspondence

| FORTRAN | ALGOL |
|---|---|
| REAL*8 | real |
| INTEGER*4 | integer |
| LOGICAL*4 | Boolean |

Note: Failure to observe these correspondences will produce unpredictable results.

4.4.1.3 Example of FORTRAN calling ALGOL

The following main program in FORTRAN has a call to a separately compiled ALGOL procedure.

```
   EXTERNAL/ALGOL/P
   DOUBLE PRECISION A(10)
   LOGICAL L,P
   DO 16 I=1,10
16 A(I)=0.0 DO
   K=2
   IF(P(A,X,K,L)) GO TO 12
   STOP
12 WRITE (6,13) A
13 FORMAT (1X,10F7.1)
   STOP
   END
```

The called ALGOL procedure is:

```
%BOOLEAN %PROCEDURE P(A,B,I,J);
%ARRAY A; %REAL B; %INTEGER I; %BOOLEAN J;
%BEGIN
   A[I] := 0.5;
%IF I > 1 %THEN P := %TRUE %ELSE P := %FALSE
%END
```

Results

The output from this program is:

```
0.0   0.5   0.0   0.0   0.0   0.0   0.0   0.0   0.0   0.0
```

Note

The special variant of the EXTERNAL statement (required because array parameters are being passed) must be the first statement in the FORTRAN program.

### 4.4.2 ALGOL calling FORTRAN

To call a FORTRAN subprogram from an ALGOL procedure it is necessary to ensure that the parameters, and, in the case of functions, function result correspond to suitable items in the program. The table in Section 4.4.2.1 lists the valid correspondence between ALGOL parameter types and FORTRAN dummy parameters. A table of corresponding function types is provided in Section 4.4.2.2.

#### 4.4.2.1 Valid Parameter Correspondence

| ALGOL | FORTRAN |
|---|---|
| integer | INTEGER*4 |
| real | REAL*8 |
| Boolean | LOGICAL*4 |
| integer array | INTEGER*4 array |
| array or real array | REAL*8 array |
| Boolean array | LOGICAL*4 array |
| integer procedure | INTEGER*4 FUNCTION |
| real procedure | REAL*8 FUNCTION |
| Boolean procedure | LOGICAL*4 FUNCTION |
| procedure | SUBROUTINE |

#### 4.4.2.2 Valid Procedure Type Correspondence

| ALGOL | FORTRAN |
|---|---|
| integer | INTEGER*4 |
| real | REAL*8 |
| Boolean | LOGICAL*4 |

Notes

1. Procedures can only be passed to a FORTRAN program unit if all the parameters of the procedure can be passed by FORTRAN.

2. Actual parameters being passed to FORTRAN must be local variables or array elements of the correct type. Expressions must be assigned to a local variable before being passed, i.e. there is no call by value. However, provided that the dummy procedure heading is followed by the pseudo-basic symbol fortran the compiler will assign an expression specified as actual parameter to a private local variable and pass an acceptable parameter.

   Note that parameters that have been passed to an ALGOL procedure by name are not local variables but subroutines and can not be passed on to FORTRAN.

### 4.4.2.3 Example of ALGOL calling FORTRAN

The following ALGOL program calls a FORTRAN subroutine FSUB which in turn calls an ALGOL procedure G

```
%BEGIN
%PROCEDURE FSUB(X,LOG,Y);
%ARRAY X;
%BOOLEAN LOG;%REAL Y;
%FORTRAN;
%INTEGER I;%BOOLEAN LOGLOG;
%REALARRAY XX[1:5];
%REAL YY;
YY:=1;
%FOR I:=1 %STEP 1 %UNTIL 5 %DO
XX[I]:=EXP(I);
LOGLOG:=%FALSE;
FSUB(XX,LOGLOG,YY);
%FOR I:=1 %STEP 1 %UNTIL 5 %DO
%BEGIN
PRINT(XX[I],0,4);NEWLINE;
%END
%END
```

The FORTRAN code is:

```
   SUBROUTINE FSUB(X,LOG,Y)
   REAL*8 X(5),Y,G
   LOGICAL LOG
   IF (LOG) RETURN
   DO 15 I=1,5
15 X(I)=G(Y)
   RETURN
   END
```

The external ALGOL procedure G required by the FORTRAN subroutine is as follows:

```
%REALPROCEDURE G(P);
%REAL P;
G:=EXP(P);
```

Results

The output from this program is:

```
2.7183 @  0
2.7183 @  0
2.7183 @  0
2.7183 @  0
2.7183 @  0
```

Notes

1. The pseudo-basic symbol <u>fortran</u> is used in the main ALGOL program just after the dummy procedure heading. It is only necessary to add the identifier of the called subroutine after the pseudo-basic symbol <u>fortran</u> when the dummy procedure name is different from the subroutine name.

2. The special variant of the EXTERNAL statement, viz.

   ```
   EXTERNAL/ALGOL/G
   ```

   is not required as no array parameters are being passed to the ALGOL procedure.

## 4.5 Calling ALGOL from IMP and IMP from ALGOL

It is possible for an IMP program to call an ALGOL procedure, and an ALGOL program to call an IMP routine.

If in an ALGOL program the compiler encounters a procedure heading whose body is the single statement <u>external</u> it will assume that an IMP routine is to be called.

The following table indicates the correspondence between IMP formal parameters and ALGOL formal parameters.

| ALGOL | IMP |
|---|---|
| integer by value | %INTEGER |
| real by value | %LONGREAL |
| Boolean by value | %INTEGER |
| integer | %INTEGERNAME |
| real | %LONGREALNAME |
| Boolean | %INTEGERNAME |
| integer array by value | No equivalence |
| Boolean array by value | No equivalence |
| real array by value | No equivalence |
| integer array | %INTEGERARRAYNAME |
| array or real array | %LONGREALARRAYNAME |
| Boolean array | %INTEGERARRAYNAME |
| procedure | %ROUTINE |
| real procedure | %LONGREALFN |
| integer procedure | %INTEGERFN |
| Boolean procedure | %INTEGERFN |
| string | %STRINGNAME |
| switch | No equivalence |
| label | No equivalence |
| No equivalence | %STRING, %RECORD |

Notes on the above table:

1. ALGOL Booleans appear to IMP as integers taking the value of 0 for false and -1 for true. *

2. Procedures can only be passed to IMP if all the parameters of the procedure can be represented in IMP.

3. Actual parameters corresponding to formal parameters being passed to IMP by name must be variables or array elements of the correct type. Expressions can only be passed to IMP by value.

## 4.5.1 Example of IMP calling ALGOL

The following IMP program calls a separately compiled ALGOL procedure, which in turn calls another ALGOL procedure:

```
%REALSLONG
   %BEGIN
      %INTEGER I,N
      %REALARRAY A(1:5)
      %OWNSTRING(20)S="IMP calls ALGOL"
      %EXTERNALROUTINESPEC CALGOL(%LONGREALARRAYNAME C,%C
      %STRINGNAME S,%INTEGERFN G,%INTEGERNAME N)
      %EXTERNALINTEGERFNSPEC G(%INTEGERNAME N)
      N=3
      CALGOL(A,S,G,N)
      %CYCLE I=1,1,5
      PRINT (A(I),4,2);NEWLINE
      %REPEAT
   %ENDOFPROGRAM
```

The called ALGOL procedure is coded thus:

```
   %PROCEDURE CALGOL(C,S,G,N);
   %ARRAY C;%STRING S;
   %INTEGER N;
   %INTEGERPROCEDURE G;%COMMENT (X):%INTEGER X;
   %BEGIN
   %INTEGER I;
   PRINTSTRING(S);NEWLINES(2);
   C[N] := G(N);
   C[N+1] := G(N+1);C[N+2] := G(N+2);
   %FOR I := 1 %STEP 1 %UNTIL N-1 %DO
   %BEGIN
   C[I] := 4.5;
   %END
   %END
```

The further ALGOL procedure referred to is provided thus:

```
   %INTEGERPROCEDURE G(X);
   %INTEGER X;
   G:=X*X*X;
```

Results

The output from the above program is:

```
   IMP calls ALGOL

     4.50
     4.50
    27.00
    64.00
   125.00
```

## 4.5.2 Example of ALGOL calling IMP

The following ALGOL program calls an independently compiled IMP routine:

```
%BEGIN
%PROCEDURE ASUB(X,N,B);
%ARRAY X; %INTEGER N; %BOOLEAN B;
%EXTERNAL;
%INTEGER I,J,K;
%BOOLEAN BB;
%REALARRAY C[1:10];
%FOR K:=1 %STEP 1 %UNTIL 10 %DO
C[K] := 0;
BB:= %TRUE;
I:=7;
ASUB(C,I,BB);
%FOR J:=1 %STEP 1 %UNTIL 10 %DO
   %BEGIN
         PRINT(C[J],0,3);NEWLINE;
   %END
%END
```

The called IMP routine is coded thus:

```
%EXTERNALROUTINE ASUB(%LONGREALARRAYNAME X,%INTEGERNAME N,B)
%INTEGER I
%IF B= -1 %THEN %START
%CYCLE I=1,1,N
X(I)=1
%REPEAT
%FINISH
%RETURN
%END
%ENDOFFILE
```

Results

The output from this program is:

```
1.000 @  0
1.000 @  0
1.000 @  0
1.000 @  0
1.000 @  0
1.000 @  0
1.000 @  0
0.000 @-99
0.000 @-99
0.000 @-99
```

Notes

1. The statement %EXTERNAL in the ALGOL program is to ensure that the compiler knows that an IMP routine is to be called.

2. Corresponding to a Boolean type parameter in the ALGOL program we have an integername type parameter in the IMP routine.

3. Because ALGOL Booleans appear to IMP as integers taking the value of 0 for false and -1 for true, the IMP statement:

   ```
   %IF B= -1 %THEN %START
   ```

   will pass control to the %START.... %FINISH block as the variable B in IMP corresponding to the Boolean variable BB in ALGOL is set to true in the ALGOL program.