



# Technical Newsletter

File Number 1410/7010-25  
Re: Form No. C28-0328-3  
This Newsletter No. N27-1269  
Date December 30, 1966  
Previous Newsletter Nos. None

IBM 1410/7010 FORTRAN

This Technical Newsletter amends the publication IBM 1410/7010 Operating System; FORTRAN, Form C28-0328-3, to include an addition concerning the SUBROUTINE subprogram.

The attached replacement pages (25-26) should be substituted for the corresponding pages now in the publication. Text changes are indicated by a vertical line to the left of the affected text.

Please file this cover letter at the back of the publication. It provides a method of determining if all changes have been received and incorporated into the publication.

*IBM Corporation, Programming Publications, Dept. 637, Neighborhood Road, Kingston, N.Y. 12401*

No card should precede the SUBROUTINE statement.

The SUBROUTINE subprogram may use one or more of its arguments to return values to the calling program. Any arguments so used must appear on the left side of an arithmetic statement or in an input list within the subprogram. The name of the SUBROUTINE must not be used as a variable in its SUBROUTINE subprogram.

The arguments may be considered dummy variable names that are replaced at the time of execution by the actual arguments supplied in the CALL statement. The actual arguments must correspond in number, order, and type to the dummy arguments.

When a dummy argument is an array name, a DIMENSION statement must appear in the SUBROUTINE subprogram. The corresponding actual argument in the CALL statement must also be a dimensioned array name.

None of the dummy arguments may appear in an EQUIVALENCE or COMMON statement in the SUBROUTINE subprogram.

Like the FUNCTION subprogram, the SUBROUTINE subprogram must return control to the calling program by a RETURN statement.

An END statement is also required.

### Subprogram Names as Arguments – The EXTERNAL Statement

Subprogram names may be used as the actual arguments in the calling program. In order to distinguish these subprogram names from ordinary variables when they appear in an argument list, their names must appear in an EXTERNAL statement (see “The Specification Statements”).

### The CALL Statement

The CALL statement is used only to call a SUBROUTINE subprogram.

General Form
CALL name (a <sub>1</sub> , a <sub>2</sub> , . . . , a <sub>n</sub> ) name is the symbolic name of a SUBROUTINE subprogram. a <sub>1</sub> , a <sub>2</sub> , . . . , a <sub>n</sub> are the actual arguments that are being supplied to the SUBROUTINE subprogram.

#### EXAMPLES

CALL MATMPY (X, 5, 40, Y, 7, 2)  
CALL QDR TIC (X, Y, Z, ROOT1, ROOT2)

The CALL statement transfers control to the SUBROUTINE subprogram and replaces the dummy variables with the value of the actual arguments that appear in the CALL statement. The arguments in a CALL statement may be any of the following: any type of constant, any type of subscripted or unsubscripted variable, an arithmetic expression, the name of a subprogram.

The arguments in a CALL statement must agree in number, order, type and array size with the corresponding arguments in the SUBROUTINE subprogram.

### Machine Indicator Tests

The 1410/7010 FORTRAN language provides machine indicator tests even though machine components referenced by the tests do not physically exist. The machine indicators, described below, are simulated by SUBROUTINE subprograms located in the System Library.

To use any of the following machine indicator tests, the user supplies the proper arguments and writes a CALL statement. In the following listing, i is an integer expression, j is an integer variable.

GENERAL FORM	FUNCTION
SLITE (i)	If i=0, all sense lights are turned off. If i=1, 2, 3, or 4, the corresponding sense light is turned on.
SLITET (i, j)	Sense light i (1, 2, 3, or 4) is tested and j is set to “1” or “2” if i is on or off, respectively. After the test, sense light i is turned off.
OVERFL (j)	This indicator is on if an arithmetic operation with real variables and constants results in an overflow condition; that is, if an arithmetic operation (of type real) produced a result whose value is greater than $(1-10^{-f}) \times 10^{99}$ . If the indicator is on, j is set to “1”; if off, j is set to “2.” The indicator is set to off after the test is made.
DVCHK (j)	This indicator is set on if an arithmetic operation with real constants and variables results in the attempt to divide by zero; j is set to “1” or “2” if the indicator is on or off, respectively. The indicator is set to off after the test is made.

#### EXAMPLES

CALL SLITE (3)  
CALL SLITET (K\*J, L)  
CALL OVERFL (J)  
CALL DVCHK (I)

As an example of how the sense lights can be used in a program, assume that the statements CALL SLITE (I) and CALL SLITET (I, KEN) have been written. Further assume that it is desired to continue with the program if sense light I is on and to write results if sense light I is off. This can be accomplished using the Relational IF statement or a Computed GO TO statement, as follows:

```

      .
      .
      IF (KEN.EQ.2) WRITE (3,26) (ANS(K), K=1, 10)
      .
      .
      .
      .
      .
      GO TO (6,17) KEN
      17 WRITE (3,26) (ANS(K), K=1, 10)
      6
      .
      .

```

**EXIT Subroutine**

A `CALL` to the `EXIT` subprogram, located in the System Library, terminates the execution of the program and returns control to the Monitor. The `EXIT` subprogram and the `STOP` statement produce identical results.

General Form
<code>CALL EXIT</code>