HOW TO WRITE AN EXTERNAL ASSEMBLY OR FORTRAN PROCEDURE

Parameters to assembly and Fortran procedures are passed in the standard OS way: a list of the parameter addresses is formed, and the address of this list is passed in register 1. The uppermost bit of the last parameter address word is set to indicate the end of the list. When the procedure is entered, R14 holds the return address, R15 has the address of the entry point (to the procedure), and R13 has the address of a save area, into which the values of the registers should be saved.

In a call on an assembly procedure, the type of each parameter is indicated by the upper byte of the parameter address word:

INTEGER : 1, REAL : 2, SHORT INTEGER : 3, LONG REAL : 4, TEXT : 6, REF(...) : 7, CHARACTER : 8, BOOLEAN : 9, LABEL : 10.

If the parameter is an array the type code is OR'ed with X'10' and if it is the last parameter it is OR'ed with X'80'. It is thus possible, within the assembly procedure, to check that the parameters are correct.

The parameter address is the variable address, the array object address (App. F), or the object code label address.

For an EXTERNAL <type> ASSEMBLY PROCEDURE, the resulting value should be loaded into register 0 (INTEGER, SHORT INTEGER, CHARACTER, BOOLEAN), floating point register 0 (REAL, LONG REAL), register 1 (REF(...)), or registers 0-2 (TEXT).

When a branch is made to a label parameter, the label address must be loaded to register 14.

An assembly procedure must always restore registers 3-14 before returning or exiting.

For a Fortran procedure call there is no type and end of parameter list indication, so it is always the responsibility of the programmer writing the call to check that the parameter list is correct. For an array parameter the address of the first element is put in the parameter list.