FORTRAN CODE SECTIONS.

SIR code can be included in Elliott FORTRAN programs when required to perform operations faster or to access basic hardware features.

A code section may be a complete sub-program or it may be part of a program with the rest written in FORTRAN. It is preceded by the word CODE on a line by itself and terminated by the word FORTRAN on a line by itself.

The statement CODE may be numbered. The label will refer to the first word in the code body. A SIR label may appear on the line before the word FORTRAN in which case the label will refer to the immediately following FORTRAN statement.

A label used in a code body must not duplicate labels used in the program or sub-program of which the code body is part. Duplication can be avoided by observance of the following rules:

1. Labels whose first letter is Q must have U as their second letter.
2. The names of variables used in the program or sub-program of which the code body is part must not be used as labels in the code body.
3. The word FORTRAN must not be used as an identifier in a code body.
4. A global name must not be located more than once.

The word FORTRAN must not be used in a comment.

A global identifier list may appear at the head of a code body if and only if the body forms a complete sub-program. The list must include the name of the sub-program, the names of any other sub-programs used in the block and, if referenced in the body, QYO, QF and QZCOM.

Patch (^), restore ($), obeyed instructions, options (*+n) sub-global labels, end-of-program (%) and halt code may not be used in a code body.

Communication with FORTRAN text.

The FORTRAN label n is equivalent to SIR label Qn where n is an integer in the range 1-99999.

Three types of variables can be used in code bodies: local variables, COMMON variables and formal parameters.

Every local variable must be located in the block in which it is used.

Common variables are not located by name, they must be accessed indirectly via the global identifier QZCOM.

Formal parameters are accessed indirectly via the local identifier QYP.

Local variables in FORTRAN text are located by the FORTRAN compiler when END is reached (for the containing program or sub-program) and are accessible to code bodies in the same program or sub-program.

If a code body is a complete sub-program all communication must be via formal parameters or COMMON variables.

Integer variables occupy a single word. Real variables occupy two words and are held in QF packed format.

Arrays are held as consecutive sequences of words. Real arrays occupy two words per element, integer arrays one word per element. Two dimensional arrays are stored with the first suffix varying most rapidly, i.e., A[1,1], A[2,1], A[3,1] ... A[2,1], A[2,2], A[2,3] ... A[3,1], A[3,2], A[3.3] ...

The FORTRAN statements

```
DIMENSION A(8), I(6,8)
B=A(M)
J=I(K,L)
A(8)=C
INT=I(4,3)
```

Correspond to the following SIR code:

```
(B=A[M])
4    M
1    M          (real numbers occupy two locations)
```

```
5    RMOD
11   QF
8    QF+1
0    RMOD
/4   A-2        (suffix count starts at 1)
5    B
0 0

(J=I[K,L])
4    L
1    -1
12   +6         ([L-1]*6)
14   17
1    K
1    -1
5    IMOD (IMODE = [L-1]*6+[K-1])
0    IMOD
/4   I
5    J

(A[8]=C)
11   QF
8    QF+1
4    C
5    A+14
0 0

(INT=I[4.3])
4    I+15
5    INT
```

If it is desired to check that subscripts are within the permitted range use should be made of the array map that contains details of every array that has been declared in a DIMENSION statement.

The map starts at location QYM. It contains consecutive three word entries stored in the order in which the arrays were declared.

The content of the first word is one of three types:

1. If the array is in COMMON, it gives the address of the first element relative to the start of the COMMON area.
2. If the array is a formal parameter, bit 14 is set to 1 and the address bits contain the address of a location in the QYP stack. This location points towards word 1 of the map of the actual parameter.

3. If the array is an actual variable, bit 14 is set to 0 and the address bits point to the first word of the array.

Bit 18 of word 2 is set to 1 if and only if the array is in COMMON.

Bit 18 of word 3 is set to 1 if and only if the array contains real elements.

The following example illustrates the structure of the map:

```
SUBROUTINE MTRXSR (R2, I, J)
DIMENSION I1(4), R1(8,8) I2(6,2)
DIMENSION R2(I,J), R3(9,10)
COMMON I2, R3

QYM  0    I1        pointer to base address of I1
     0    4         dimension 1
     0    0         dimension 2 = 0
     0    R1        pointer to base address of R1
     0    8         dimension 1
     /0   8         dimension 2; with real indicator
     +0             offset of I2 in COMMON area
     /0   6         dimension 1; with COMMON indicator
     0    2         dimension 2
     1    QYP+0     indirect pointer to actual map for
                    formal parameter R2
     0    0         dummy dimension
     /0   0         dummy dimension; with real indicator
     +12            offset of R3 in COMMON area
     /0   9         dimension 1; with COMMON indicator
     /0   10        dimension 2; with real indicator
```

The follow example illustrates addressing COMMON variables. The FORTRAN code

```
DIMENSION ARR(3,4)
COMMON X, I, ARR
Y=X
I=2
ARR(J,K)=Y
Z=ARR(1,2)
```

can be reduced to the following code body:

```
     11   QF
     8    QF+1
     0    QZCOM
     /4   0          (Y=X)
     5    Y
     +0
     4    +2         (I=2)
     0    QZCOM
     /5   2

     4    K          (ARR[J,K]=Y)
     1    -1
     12   +6
     14   17
     1    J
     1    J
     1    -2
     1    QZCOM
     5    R
     11   QF
     8    QF+1
     0    R
     4    Y
     /5   3
     0    QZCOM
     /4   9
     5    Z
     0 0
```

A sub-program may be written entirely in SIR code. If, however, an executable FORTRAN statement is included in the sub-program, then the sub-program head and any specification statements must also be written as FORTRAN text.

The code equivalent to the FORTRAN statement

```
     FUNCTION FUNC(I, R2, A, AP, P)
```

where A, AP are real arrays, I is an integer and P, R1, R2 are real numbers is:

```
[FUNC QF QYO QZCOM]
FUNC +0             (link)
     11   QYO       (get actual parameters)
     8    QYO+1     (and place their addresses in QYP et seq)
     +5             (number of parameters)
     &360000        (real/integer display)
     +0             (space for array display)
```

```
QYP  +0            (actual parameter pointers)
     +0
     +0
     +0
     +0
     +0
```

If a formal parameter is an array, the QYO routine places the address of the first word of its map in the appropriate QYP location.

The last word of the QYP stack is used to hold the address where the result is to be stored, in the case of a FUNCTION sub-program.

In the integer/real display, bit 18 is 1 if parameter 1 is real, otherwise 0 if the parameter is an integer or non-existent, bit 17 is 1 if parameter 2 is real and so on.

In the array display bit 18 is 1 if parameter 1 is an array, otherwise 0 if the parameter is a simple variable or non-existent. Bit 17 is a 1 if parameter 1 is an array and so on.

If an actual parameter in the QYP stack is also a formal parameter of the calling sub-program, bit 14 of the corresponding pointer is set to 1 and the address bits point to where the actual address may be found. (QYO will follow back through a chain of multiple such references if required to determine the ultimate location for the parameter.) Otherwise if bit 14 is 0, the address bits point to the actual parameter, or the map of the actual parameter as appropriate.

Floating Point Arithmetic.

QF and QFMATH may be used as described in the section on SIR programming, however following names must be used in code bodies for reference to QF and QFMATH:

Name used in SIR	Name used in FORTRAN code bodies
QF	QF
LN	QFLOG
EXP	QFEXP
SQRT	QFSQRT

SIN	QFSIN
COS	QFCOS
ARCTAN	QFATAN

The library functions may be called the same way as any other sub-program. In the 16K(LP) system the labels QFLOG, QFEXP, QFATAN etc are not available for use by CODE statements and the library function entries must be used.

Input and Output.

Using the same variables as the previous example, the FORTRAN statement

```
WRITE (2,999) A, P, AP, I, R1
```

is equivalent to the following code:

```
11   QYO          (entry to input/output package)
8    QYO+2
/0   +2           (write indicator and channel number)
0 Q999+1          (pointer to FORMAT code)
+5                (number of parameters)
&720000           (real/integer display)
&500000           (array display)
0 QYM+0           (pointer to map of A)
1 QYP+0           (indirect pointer to P)
1 QYP+1           (indirect pointer to map of AP)
0 I               (pointer to I)
0 R1              (pointer to R1)
```

Note the sign bit of the first word after the subroutine entry is set to 1 if and only if the statement is a WRITE statement. The address points to a location containing the channel number.