Procedure

The subroutine DEC is used to set up a block if this is the first declaration in the block, a stopper is put in the name list, the marker PH is set to show that a procedure heading is being processed, and a call is made to BCR to read the next ALGOL section i.e. 'procedure name' (. A check is made that the preceding delimiter was real integer, Boolean or non-type. Unless the procedure is own code an unconditional jump is compiled to jump round the procedure body and a stack entry is made of proc. begin together with various state variables and the address (PP) of the incomplete UJ operation to be updated at the end of the procedure body.

DECSTA is set to statement level for the procedure statement following the specification part, the number of parameters (PRMCOU) is set to zero and the current block name (BN) is updated for the procedure block.

If the procedure has no parameters (delimiter is;) DECTYP is set to procedure zero and the procedure name is declared in the namelist using the subroutine DECL. The procedure entry operation is compiled, DECTYP cleared and a call is then made to BCR to check whether the procedure body is ALGOL or own code.

If the procedure has parameters, DECTYP is set and the procedure name is declared in the namelist. PRCENT is used to hold the namelist address of the procedure name, and PROC is set to 1. The loop that follows reads a formal parameter, checks that it has not already been declared, enters it in the namelist and updates the count of parameters. When ')' has been read, the procedure entry operation can be compiled and the namelist entry completed with the number of formal parameters; PROC and DECTYP are cleared.

The next call to BCR should fetch ; which terminates the formal parameter part.

Example:

```
                                  f.p. part     value part
integer procedure  F (a, b, c) ;  value a,b;

                                  specification part
                                  real a,c;     array b;
```

Another call is made to BCR to determine whether a value part or the specification part follows. Each identifier in the value part is checked for appearance in the formal parameter list. v:=1 in the namelist entry. When the terminating ';' is read the next ALGOL section is read (specifier,) and the specification part is processed.

This consists of specifiers followed by identifier list(s) e.g. real a,c; There is an inner loop to read each identifier in the list following the specifier, check that it is a formal parameter and complete the namelist entry. If the formal parameter is a switch the number of dimensions is set to 1 but if it is an array or procedure

the number of dimensions or parameters is not yet known and is set to + 15, This number will be updated at the first occurence of this identifier with its parameters, and subsequent occurences must agree.

If the delimiter is ';' this could be the end of the specification part or the end of this specifier list. In the former case, a check is now made that each formal parameter has been specified and if called by value that the type is not switch, string or procedure. The program then compiles the check words for the run time parameter checking.

Finally, a check is made to determine whether the program body is ALGOL or own code. If the latter DECSTA is reset to declaration level, the formal parameters are collapsed and the next delimiter is read to discard the ; 903 ALGOL requires the parameters of formal procedures to be called by name, this is checked in PRAMCH however, when a parameter is found to be of type procedure.

ERRORS

FAIL 101 ; procedure name not followed by ; or (

FAIL 102 ; formal parameter part not followed by ;

FAIL 90 ; wrong delimiter in value or specifier part

FAIL 109 ; constant not allowed in procedure heading

FAIL 65 ; illegal specifier

FAIL 17 ; identifier in specification part is not a formal parameter, or formal parameter not fully specified

FAIL 94 ; string, switch or procedure called by value

FAIL 6 ; more than 14 parameters

FAIL 86 ; procedure inside another declaration

FAIL 85 ; Name list overflows

FAIL 92 ; Identifier not specified

FAIL 88 ; formal parameter not followed by) or ,