CGTM # 23
Ira Pohl
September, 1967

Translation from B5500 Extended Algol to OS/360 Algol

Introduction

OS/360 Algol is the IBM implementation of Algol 60[1] for the System/360[2]. Extended Algol[3] for the Burroughs B5500 contains many more capabilities than Algol 60 but is defined analogously. Therefore, the translator from B5500 Algol to OS/360 Algol is a mapping from a superset into a (approximately) proper subset. It is not a very hard translation task for the elements contained in the intersection of the two languages, and is very hard for some of the machine dependent extensions of B5500 Algol. The translator was written to be a simple translator dealing with translating the Algol 60 structures of B5500 Algol into OS/360 Algol.

The translation process handles simple translations which are equivalent to finite state transformations. This level of translation is the simplest possible. It would be harder to deal with Fortran to PL/1 which requires context free transformations, and harder still to go from B5500 Algol to PL/1 which would require dealing with problems of semantic equivalence of similar syntactic form. The translator described here for B5500 Algol to OS/360 Algol mostly does work akin to the following production, <reserved word> ::= '<quoted reserved word>'. A Fortran to PL/1 [4] translator deals in harder syntactic transformations where, for example, the Fortran <do statement> must be translated with an END inserted at the place of the statement number that ends the <do statement>. Harder still[5] is to appropriately translate a B5500 Algol <for statement> into a PL/1 <do statement> where the syntax does not differ greatly but the meaning does. The Algol statement constantly

re-evaluates its step size and upper bound, but the PL/1 statement does this only once. For reasonable translation here, it becomes necessary to translate the Algol <for statement> into the corresponding conditional and branching statements, or to check that the Algol statement being executed in the for loop is not affecting the step size or bound.

Use

The translator is simple to use, and its output requires little further modification. The translator takes as input a B5500 deck (extended IBM 026 key punch code) and produces an OS/360 Algol translation[6]; this translation is produced as a sequenced deck (IBM 029 key punch code) and for input to the OS/360 Algol compiler. The translator deletes certain constructions which would cause the OS/360 Algol compiler undue syntactic hardship. The translator produces a copy of these deletions referencing the sequence numbers of the outputted deck, and the OS/360 Algol compiler produces its standard syntactic diagnostic messages.

Further translation by hand (see Appendix 1) is required in certain areas, principally input/output. If stream procedures and bit manipulation are extensively used the program should probably not be translated, but reprogrammed and not necessarily in Algol. However, if the B5500 Algol program mainly uses the Algol 60 subset of Extended Algol, then the hand translation will normally be limited to input/output. The (human) translator should be familiar with both B5500 Algol and OS/360 Algol and should be sure to run a few test cases on the translated program.

Organization

Paralleling the simple nature of the translation is the simple organization of the translation. Abstractly it is a finite state recognizer with left recursion, needing no backup in its left-to-right scan.

The program is written in OS/360 Algol, which requires that string manipulation be carried out as the manipulation of integer arrays. While this is unnatural (as opposed to the natural handling of strings in PL/1) the experience gained in using the OS/360 Algol system, and the excellent compile time and especially run time diagnostic capability of OS/360 Algol were sufficiently compensatory.

The program translates a card at a time, reading each source card into an integer array. This integer encoding is then scanned for one of four basic elements -- identifier, reserved word, number or special symbol (which includes blanks). Identifiers and numbers are directly passed out in 029 card code form. Reserve words and special symbols require a multiplicity of actions and each is done by a procedure having the various relevant cases.

The special symbols are mapped into one or more output symbols. The most complicated case is the quote mark (') which initiates a scan for a second quote mark, meanwhile directly emitting all characters encountered. The reserved words are more diverse and complicated in their requirements. The simplest action is to emit them quoted -- so BEGIN becomes 'BEGIN'. Other reserved words initiate simple mappings into special symbols, as is the case with OR, AND and NOT going respectively to |, &, and ¬. Certain reserved words and the statement containing them are deleted. Label declarations are unnecessary, while input/output statements are intractable, so both are deleted. FINØFTRAN is considered a reserved word and its detection causes termination of the translation. The translation is both punched on cards and passed to the OS/360 Algol compiler. The compiler then complains about any text it cannot parse.

The translator is very simple in structure and is easily modified. Extensions could be made in two manners, first as prepasses to this

system or second as more cases in the handling of reserved words. A prepass could be written to handle the expansion of DEFINE'S, FILL'S and the translation of the WHILE and FOR-STEP-WHILE which are not found in OS/360 Algol. These constructions require keeping track of block structure, tasks unsuitable for the present translator. The second possibility, extending the cases in handling the reserve words, could in a more limited way than the above method, add scope to the translator. The extensions could include the deletion of lower bounds lists from array specifications. Nevertheless, the translator as it stands should allow a programmer familiar with both B5500 Algol and OS/360 Algol to translate and debug a 1000 card program in a day or two providing the program is written principally in the Algol 60 subset of B5500 Algol.

Acknowledgements

I would like to thank John Welsch for his critical attention, advice and moral support. Also, would I like to thank Miss Kathleen Maddern for her excellent error detection and typing.

Footnotes and References

1. 'Revised Report on the Algorithmic Language ALGOL 60', CACM Vol. 6 (1963) p. 1.

2. 'IBM System/360 Operating System ALGOL Language', C28-6615-1.

3. 'Burroughs B5500 Extended Algol Reference Manual', 1964.

4. Lanse Leach, 'FORTRAN to PL/1 Translator', Stanford CC, Aug, 1967, Document No. 33-78-2.

5. The key is that semantic equivalence is harder to achieve going from B5500 Algol to PL/1 than it is going from Fortran to PL/1. One is misled by the syntactic differences in judging Fortran the harder translation. For the opposite view see reference 4.

6. The extended 59 character set is used instead of the ugly 48 character set. See appendix 2 of reference 2 (p. 94).

See also: Wayne Wilner, Algol to Fortran Translator, CGTM 15, June 1967

APPENDIX 1 HAND TRANSLATION TIPS

1) 'FOR' V := AE1 'STEP' AE2 'WHILE' BE 'DO'
statement;

Here AE stands for arithmetic expression, BE stands for Boolean expression, and V for variable.

In translating this constuction a new variable is declared and used. New variables will be represented by Q, Q1, Q_2 . . .

Hand translation:

```
    Q:= AE 1;
'FOR' V := Q 'WHILE' BE 'DO'
'BEGIN'   statement;  Q := Q + AE2 'END';
```

2) 'WHILE' BE 'DO' statement;

Hand translation:

'FOR' Q := 0 'WHILE' BE 'DO' statement;

3) FILL A[i_1 , i_2 , .. , *] WITH value 1, value 2, ... value n;

Value i are the values in the value list, and in B5500 Algol these may be numbers, strings or octal numbers. In the case of strings or octal numbers a simple translation may not be possible.

Hand translation:

```
    J := lower bound of row;
'FOR' I := value 1, value 2, ... value n  'DO'
  'BEGIN'  A(/ i1, i2, .. , J/) := I; J:= J+1 'END';
```

Note: A, I,and value 1, value 2, ... should agree as to type.

4) X MOD Y

Hand translation:

X - Y * (SIGN(X/Y) * ENTIER (ABS(X/Y)))

5)
```
CASE AE OF
  'BEGIN'
     S1;
     S2;
     :
     Sn;
  'END';
```

S1 through Sn represent statements. In the following translation each CASE statement must have its own switch list.

Hand translation:

```
'BEGIN'
    'SWITCH'  CASE := CASE 1, CASE 2, ... , CASE n;
      'GO TO'  CASE (/AE/);
CASE 1:  S1; 'GO TO' EXIT;
CASE 2:  S2; 'GO TO' EXIT;
  '
  '
  '
CASE n:  Sn;
  EXIT:
  'END';
```

6) 'IF' BE 'THEN' conditional statement;

Hand translation:

```
'IF' BE 'THEN'  'BEGIN' conditional statement 'END';
```

7) Own variables

These should be moved to the outer block, while consistently changing any conflicts between the scopes of identical identifiers.

8) ALPHA variables

The translator changes these to REAL, but normally this will not suffice. A simple translation may not be possible.

9) Lower bounds in array specifications must be removed.

```
   'PROCEDURE'  P(A);
      'ARRAY'   A(/0/);
```

Hand translation:

```
      'PROCEDURE' P(A);
           'ARRAY'  A;
```

10) Double statements

If double precision is required the LONG option should be used when calling the OS/360 Algol compiler. Mixing LONG and SHORT is impossible in OS/360 Algol.

11) Machine dependent constructions - stream procedures, partial words etc.

These create difficult reprogramming problems.

12) Input/output

Read the relevant sections of the Algol manual.

If the above tips do not lead to a painless conversion, try going to Fortran or PL/1 for real difficulty.

```
 SC      SOURCE STATEMENT

00000       'BEGIN' 'COMMENT'  B5500 ALGOL TO OS/360 ALGOL TRANSLATOR.
00000          PROGRAMMER  IRA POHL  (SEAC 650)  COMPLETED SEPTEMBER , 1967.
00000             THE FOLLOWING PROGRAM WILL TRANSLATE A B5500 EXTENDED
00000          ALGOL PROGRAM TO OS/ALGOL. THE TRANSLATION PRODUCES A PUNCHED
00000          DECK AND ALSO A FIRST ATTEMPT BY THE OS/ALGOL SYSTEM TO COMPILE
00000          THE TRANSLATED PROGRAM.  SINCE OS/ALGOL IS PURE ALGOL 60 WITH
00000          FEW DIFFERENCES THE TRANSLATOR HANDLES THIS SUBSET OF B5500
00000          ALGOL VERY WELL LEAVING LITTLE FOR HAND CONVERSION. IO,STREAM
00000          PROCEDURES, B5500 WORD OPERATIONS AND OTHER EXTENSIONS WILL
00000          BE FLAGGED BY THE COMPILER OR DELETED.              ;
00000
00000
00000       'COMMENT'  A,B,C,...Z  =GIVEN INTEGER VALUE FOR INSYMBOL/OUTSYMBOL
00000          AFTER INITIALIZATION OF RESERVED WORD TABLE H,I,J,K ARE USED
00000          FOR INTEGER VARIABLES AND MAY NO LONGER BE ASSUMED TO HAVE
00000          SYMBOLIC USE.
00000          NUMCRD=  COUNTER FOR NUMBER OF CARDS TO BE OUTPUTTED PER
00000          CARD INPUT. CURRENT MAXIMUM IS 3.
00000          CRDCNT= CARD BEING PROCESSED .
00000          CCRDCNT= CARD NUMBERING PRODUCED BY SEQUENCE.
00000          CODE1= ENTRY IN RESERVED WORD TABLE. IT IS USED TO INDEX MPO
00000          WHICH ALLOWS SWITCH TO APPROPRIATE CASE IN RWORDS.
00000          CODE=  SWITCH FOR TRANSLATE ROUTINE  1=RESERVED WORD
00000             2= IDENTIFIER    3= NUMBER     4= SPECIAL SYMBOL
00000          CP= COLUMN POINTER FOR CURRENT CARD
00000          COR= COLUMN POINTER FOR OUTPUT CARD
00000          BUF(/1:80/)= INPUT BUFFER   CRD(/1:3,1:80/) =OUTPUT BUFFER
00000          RWD= RESERVED WORD TABLE. IT IS LINEAR AND ORDERED BY LENGTH
00000          OF THE RESERVED WORDS.  TABLE(/I/) GIVES THE START OF THE TABLE
00000          FOR RESERVED WORDS OF LENGTH I. ENTAB(/I/)= END OF TABLE .
00000          PREV=BOOLEAN WHICH SAYS WHETHER CURRENT SYMBOL HAS BEEN
00000          TRANSLATED IF TRUE THEN NEW SYBOL IS READ.
00000          MAP(/I/)= MAPPING FOR SPECIAL SYMBOLS  MPL=MAPPING FOR RESERVED
00000          WORDS.                          ****************   ;
00000       'INTEGER' A,B,C,D,E,F,G,L,M,N,O,P,Q,R,S,T,U,W,V,X,Y,Z;
00001       'INTEGER' H,I,J,K,MAXLEN,WORMAX,SYMB,BLANK,LETORDIG,DIGIT,CP,COR,
00001          CCRDIT,IDEN,NUMB,SYMCOD,LENGTH,CODE;
00002       'INTEGER' NUMCRD,CRDCNT, CODE1,CCRDCNT;
00003       'INTEGER' 'ARRAY' BUF(/1:80/), CRD(/1:3,1:80/), TABLE,ENTAB(/1:9/),
00003          RWD(/0:1000/),  ITEM(/0:300/);
00004       'INTEGER' 'ARRAY' MAP(/0:63/), MPD(/0:500/);
00005       'BOOLEAN' PREV;
00006
00006       'COMMENT' NL(I) SKIPS I LINES ON PRINTER    **********  ;
00006       'PROCEDURE' NL(I);  'VALUE' I; 'INTEGER' I;  SYSACT(1,14,I);
00010
```

```
 SC     SOURCE STATEMENT

0001C       'COMMENT'  SEQUENCE PLACES SEQUENCE NUMBERS ON CARDS.********    ;
0001C       'PROCEDURE' SEQUENCE;
00011       'BEGIN'  'INTEGER' I,K;'INTEGER' 'ARRAY' DIG(/1:8/);
00013          K:=CORDCNT;
00014          DIG(/8/):=0;
00015          DIG(/1/):= CORDCNT '/'1000000;CORDCNT:=CORDCNT-DIG(/1/)*1000000;
00017          DIG(/2/):= CORDCNT '/'100000; CORDCNT:=CORDCNT-DIG(/2/)*100000;
00019          DIG(/3/):= CORDCNT '/'10000;  CORDCNT:=CORDCNT-DIG(/3/)*10000;
00021          DIG(/4/):= CORDCNT '/'1000;   CORDCNT:=CORDCNT-DIG(/4/)*1000;
00023          DIG(/5/):= CORDCNT '/'100,    CORDCNT:=CORDCNT-DIG(/5/)*100;
00025          DIG(/6/):= CORDCNT '/' 10  ;  CORDCNT:=CORDCNT-DIG(/6/)*10  ;
00027          DIG(/7/):= CORDCNT;
00028          'FOR' I:=1 'STEP' 1 'UNTIL' 8 'DO'
00028          'BEGIN'
00029             OUTSYMBOL(2, '('0123456789')', DIG(/I/)+1);
00029             OUTSYMBOL(3, '('0123456789')', DIG(/I/)+1);
00030          'END';
00031          CORDCNT:=K+1
00031       'END' SEQUENCE;
00032
00032       'COMMENT' OUTCRD PRODUCES PUNCHED CARDS  DATA SET NUMBER 2
00032          AND PASSES OUTPUT TO ALGOL COMPILER DATA SET 3. ***** ;
00032       'PROCEDURE' OUTCRD    (NCD); 'INTEGER' NCD;
00034       'BEGIN'  'INTEGER' I;
00035          'FOR' I:=1 'STEP' 1 'UNTIL' 72 'DO'
00035          'BEGIN'
00035             OUTSYMBOL(2,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/=<>¬&()
00035  :';&|?  .$,#_'&')',CRD(/NCD,I/) );
00036             OUTSYMBOL(3,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/=<>¬&()
00036  :';&|?  .$,#_'&')',CRD(/NCD,I/) );
00037          'END';
00038       SEQUENCE;   'COMMENT'  SEQUENCE NUMBERS ON CARDS.    ;
00039       'FOR' I:=1 'STEP' 1 'UNTIL' 80 'DO'  CRD(/NCD,I/):=0;
00040          COR:=1;
00041       'END'  OUTCRD;
00042
00042
00042       'COMMENT' INCRD READS IN CARDS IN 026 AS REQUESTED. CRDCNT IS THE
00042          NUMBER OF CARDS READ IN. FOR EACH CARD READ IN INCRD CALLS
00042          OUTCRD TO GENERATE THE PREVIOUS CARDS TRANSLATION. IF THE
00042          PREVIOUS CARD HAS TRANSLATED INTO TWO CARDS THE OVERFLOW IS
00042          HANDLED BY STARTING A SECOND CARD AT COLUMN ONE.  ;
00042       'PROCEDURE'  INCRD;
00043       'BEGIN'  'INTEGER' I,K;
00044       'COMMENT'  CHECK FOR BLANK CARD        **********       ;
00044       'FOR' K:=1 'STEP' 1 'UNTIL' 80 'DO'
00044          'IF' CRD(/NUMCRD,K/)¬=0 'THEN' ' GO TO' L1;
00045          NUMCRD:=NUMCRD-1;
```

```
 SC      SOURCE STATEMENT

0004C      L1:
00046        'FOR' K:=1 'STEP' 1 'UNTIL' NUMCRD 'DO' OUTCRD(K);
00047           'FOR' I:=1 'STEP' 1 'UNTIL' 80 'DO'
00047           'BEGIN'
00047              INSYMBOL(0,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ&-=/#+    %<'
00047    a;_>)(*.$,¬"?|')',BUF(/I/) );
0004C           'END';
00049           NUMCRD:=1;
0005C        CRLCNT:=CRLCNT+1;
00051        'END' INCRD;
00052
00052        'COMMENT' NXTSYM GETS NEXTSYMBOL FROM INPUT BUFFER BUF .******  ;
00052        'INTEGER' 'PROCEDURE' NXTSYM,
00053        'BEGIN'
00053           'IF' CP=73 'THEN'
00053           'BEGIN' CP:=1;  INCRD, 'END';
0005C           NXTSYM:=BUF(/CP/);  CP:=CP+1;
00058        'END'  NXTSYM;
00059
00059        'COMMENT' INTGER RECOGNIZES IDENTIFIERS.*****     ;
00059        'PROCEDURE' INTGER(LENGTH,ITEM);
00060           'INTEGER' LENGTH; 'INTEGER' 'ARRAY' ITEM;
00062        'BEGIN'
00062           'INTEGER' K;
00063           K:=1;
00064           LENGTH:=1;  SYMB:=NXTSYM;   PREV:='FALSE';
00067           'FOR' LENGTH:=LENGTH+1  'WHILE'  ¬((SYMB=BLANK) | (SYMB>LETORDIG)
00067     ) 'DO'     'BEGIN'   ITEM(/LENGTH/):=SYMB;
00068                    SYMB:=NXTSYM;
00069           K:=LENGTH;
0007C                'END';
00071           LENGTH:=K;
00072        'END' INTGER;
00073
00073        'COMMENT' NUMEND IS TRUE IF SYMBOL MAY APPEAR IN A NUMBER. IT
00073           IS USED BY INNUMB TO FIND THE END OF A NUMBER. *****  ;
00073        'BOOLEAN' 'PROCEDURE' NUMEND;
00074           'IF' SYMB<=LIGIT & SYMB¬= BLANK 'THEN' NUMEND:='TRUE' 'ELSE'
00074           'IF' (SYMB=37 | SYMB=38) | (SYMB=49 | SYMB=50) 'THEN'
00074              NUMEND:='TRUE'  'ELSE' NUMEND:='FALSE';
00075
00075        'COMMENT' INNUMB RECOGNIZES NUMBERS.  **************     ;
00075        'PROCEDURE' INNUMB(LENGTH,ITEM),
0007C           'INTEGER' LENGTH; 'INTEGER' 'ARRAY' ITEM;
0007E        'BEGIN'
0007E           'INTEGER' K;
00079           K:=1;
0008C           LENGTH:=1;  SYMB:=NXTSYM;   PREV:='FALSE';
```

```
SC      SOURCE STATEMENT

0C083           'FOR' LENGTH:=LENGTH+1 'WHILE' NUMENC 'DO'
0C084           'BEGIN' ITEM(/LENGTH/):=SYMB; SYMB:=NXTSYM; K:=LENGTH 'END';
0C086           LENGTH:=K;
0C087       'END' INNUMB;
0C088
0C088       'COMMENT' OUTIDEN OUTPUTS IDENTIFIERS ,BUT ACTUALLY MAY OUTPUT
0C088          ANY QUANTITY IN ITEM ARRAY.   ************             ;
0C088       'PROCEDURE' OUTIDEN(LENGTH,ITEM);
0C089          'VALUE' LENGTH;  'INTEGER' LENGTH;  'INTEGER' 'ARRAY' ITEM;
0C092       'BEGIN'  'INTEGER' I,J,LAS;
0C093          LAS:= COR+LENGTH-1;
0C094          J:=0;
0C095     NWC:
0C095          'IF' LAS>72 'THEN'
0C095             'BEGIN'
0C095                      'FOR' I:=COR 'STEP' 1 'UNTIL' 72 'DO'
0C095                'BEGIN' J:=J+1; CRD(/NUMCRD,I/):=ITEM(/J/) 'END';
0C097                NUMCRD:=NUMCRD+1;
0C098                LAS:=LENGTH+COR-73, COR:=1;  'GOTO' NWC;
0C101             'END';
0C102          'FOR' I:= COR 'STEP' 1 'UNTIL' LAS 'DO'
0C102                'BEGIN' J:=J+1; CRD(/NUMCRD,I/):=ITEM(/J/) 'END';
0C104       COR:=LAS+1;
0C105       'END' OUTIDEN;
0C106
0C106       'COMMENT'   OUTNUM OUTPUTS NUMBERS          ***********    ;
0C106       'PROCEDURE' OUTNUM (LENGTH,ITEM);
0C107          'VALUE' LENGTH;  'INTEGER' LENGTH;  'INTEGER' 'ARRAY' ITEM;
0C110          OUTIDEN(LENGTH,ITEM);
0C111
0C111       'COMMENT' OUTSYM  OUTPUTS SYMBOLS           ********   ;
0C111       'PROCEDURE' OUTSYM(SYMB),  'VALUE' SYMB; 'INTEGER' SYMB;
0C114       'BEGIN'  'COMMENT' NEED LARGE NUMBER OF CASES   ;
0C114       'IF' COR>72 'THEN' 'BEGIN' NUMCRD:=NUMCRD+1; COR:=1 'END' ;
0C116       CRD(/NUMCRD,COR/):=SYMB;  COR:=COR+1;
0C118       'END' OUTSYM;
0C119
0C119       'COMMENT'OITEM OUTPUTS ITEM INDENTIFIER ROUTINES*****;
0C119       'PROCEDURE' OITEM(LENGTH,ITEM);
0C120       'VALUE' LENGTH; 'INTEGER' LENGTH; 'INTEGER' 'ARRAY' ITEM;
0C123       'BEGIN'  'INTEGER' I,J;
0C124       'FOR' I:=1 'STEP' 1 'UNTIL' LENGTH 'DO'
0C124            OUTSYMBOL(I,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/=<>¬&(
0C124  :';&|?  .$,#_'%')',ITEM(/I/) );
0C125       'END';
0C126
0C126       'PROCEDURE' OUTDELETE(MESSAGE,NSYM,ITEM);
0C127          'STRING' MESSAGE;  'INTEGER' NSYM;  'INTEGER' 'ARRAY' ITEM;
```

```
SC      SOURCE STATEMENT

00130           'COMMENT'  USED TO PRINT OUT DELETED MATERIAL. THE DELETED
00130           MATERIAL IS NSYM LONG AND IS FOUND IN ITEM. A MESSAGE IS ALSO
00130           ISSUED.    ;
00130         'BEGIN'      'INTEGER'  I;
00131           NL(3);  OUTSTRING(1,'(***  ')')    ;
00133                     OUTSTRING(1,MESSAGE); OUTSTRING(1,'(' CARD')');
00135           OUTINTEGER(1,CARDCNT),      NL(1);
00137           'FOR' I:=1 'STEP' 1 'UNTIL' NSYM 'DO'
00137              OUTSYMBOL(1,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/=<>¬&()
00137   :';&|?  .$,#_'<')',ITEM(/I/) );
00138         'END'  OUTDELETE;
00139
00139         'COMMENT' SCANTILL SCANS AND DELETES UNTIL CHARACTER TIL.****** ;
00139         'PROCEDURE' SCANTILL(TIL); 'VALUE' TIL;  'INTEGER' TIL;
00142         'BEGIN'
00142       CONTSCAN:
00142              OUTSYMBOL(1,'('0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ+-*/=<>¬&()
00142   :';&|?  .$,#_'<')',SYMB );
00143           SYMB:=NXTSYM;
00144           'IF' SYMB =: TIL 'THEN' 'GO TO' CONTSCAN;
00145           PREV:='TRUE';      'COMMENT' DELETION INCLUDES TIL ;
00146         'END' SCANTILL;
00147
00147         'COMMENT' SCANTILL SCANS UNTIL CHARACTER TIL AND PLACESSYMBOLS
00147           DIRECTLY IN OUTPUT BUFFER. IT IS USED FOR STRINGS AND COMMENTS.;
00147         'PROCEDURE' SCANTILL(TIL); 'VALUE' TIL;  'INTEGER' TIL;
00150         'BEGIN'
00150       CONTSCAN:
00150           OUTSYM(SYMB);
00151           SYMB:=NXTSYM;
00152           'IF' SYMB =: TIL 'THEN' 'GO TO' CONTSCAN;
00153           PREV:='TRUE';
00154         'END' SCANTILL;
00155
00155         'COMMENT' FINDS END,ELSE OR ;
00155         'PROCEDURE' SCANENDORELSE;
00156         'BEGIN'
00156       RECHECK:
00156           NXTITEM(LENGTH,ITEM,CODE);
00157           'IF' CODE=4 'THEN' 'BEGIN' 'IF' ITEM(/1/)=50 'THEN'
00157              'BEGIN' OUTSYM(50); 'GO TO' FOUNDEND  'END'
00158              'END'
00158              'ELSE' 'IF' CODE=1 'THEN' 'BEGIN'
00158                       'IF' CODE1=13 | CODE1=49 'THEN'
00158                       'BEGIN' BACKUP(LENGTH,ITEM,CODE1); 'GO TO' FOUNDEND
00159                       'END'
00159                     'END' ;
00160           CITEM(LENGTH,ITEM);
```

```
SC      SOURCE STATEMENT

00161           'GO TO' RECHECK;
00162       RECHECK:
00163         'END' SCANENDOFSTAT;
00164
00165         'COMMENT' RSWWORD SEARCHES RESERVED WORD TABLE FOR IDENTIFIER
00166            IN ITEM AND IT RETURNS CODE1 THE POSITION IN TABLE IF THE
00167            IDENTIFIER IS A RESERVED WORD.    *********              ;
00168         'BOOLEAN' 'PROCEDURE' RSWWORD(ITEM,LENGTH);
00169         'VALUE' LENGTH;     'INTEGER' LENGTH;'INTEGER' 'ARRAY'  ITEM;
00170         'BEGIN'  'INTEGER' I,J,K,LAST;
00171            'IF' LENGTH<MAXLEN 'THEN'
00172            'BEGIN'
00173               'FOR' K:=TABLE(/LENGTH/) 'STEP' LENGTH 'UNTIL'
00174                  ENTAB(/LENGTH/) 'DO'
00175               'BEGIN'
00176                  CODE1:=K;
00177                  LAST:=K+LENGTH-1;  I:=0;
00178                  'FOR' J:=K 'STEP' 1 'UNTIL' LAST 'DO'
00179                  'BEGIN'  I:=I+1;
00180                     'IF' RWD(/J/) =/= ITEM(/I/) 'THEN' 'GOTO' NEWID
00181                  'END';
00182                  RSWWORD:= 'TRUE',  'GOTO' ROUT;
00183       NEWID:  'END'
00184            'END' ;   RSWWORD := 'FALSE';
00185       ROUT:
00186         'END' RSWWORD;
00187
00188         'COMMENT' SPECSYM  PERFORMS APPROPRIATE TRANSLATION OF THE
00189            SPECIAL SYMBOLS.  ********          ;
00190         'PROCEDURE' SPECSYM(CHR);
00191            'VALUE' CHR; 'INTEGER' CHR,
00192         'BEGIN'
00193            'SWITCH' CASE:= CASE1,CASE2,CASE3,CASE4,CASE5,CASE6,CASE7,
00194                CASE8,CASE9,CASE10;
00195         'GO TO' CASE(/ MAP(/CHR/) /),
00196        CASE1:  OUTSYM(44); OUTSYM(41);  'GO TO' NWITEM; 'COMMENT' =/= ;
00197        CASE2: OUTDELETE('(' % ENCOUNTERED REST OF CARD DELETED ')',0,ITEM);
00198            CP:=73;  'GO TO' NWITEM;  'COMMENT'  PERCENT ENCOUNTERED      ;
00199        CASE3:      OUTSYM(49); OUTSYM(40); OUTSYM(49); SYMB:=NXTSYM;
00200            SCANTILL(49);
00201         OUTSYM(49); OUTSYM(47); OUTSYM(49);   'GO TO' NWITEM;
00202            'COMMENT' CASE 3 QUOTES AND PASSES STRINGS   *******  ;
00203        CASE4:   OUTSYM(48); OUTSYM(41);  'GO TO' NWITEM; 'COMMENT'  :=   ;
00204        CASE5:   OUTSYM(42); OUTSYM(41);  'GO TO' NWITEM; 'COMMENT' <=   ;
00205        CASE6:   OUTSYM(40); OUTSYM(47);  'GO TO' NWITEM; 'COMMENT'  /)  ;
00206        CASE7:   OUTSYM(46); OUTSYM(40);  'GO TO' NWITEM; 'COMMENT' (/  ;
00207        CASE8:  OUTSYM(SYMB); 'GO TO' NWITEM;  'COMMENT'  STANDARD 1-1 MAP ;
00208        CASE9:  OUTSYM(39);  OUTSYM(39),   'GO TO' NWITEM; 'COMMENT'  **   ;
```

```
 SC     SOURCE STATEMENT

0C215       CASE15:     QUTSYM(43); QUTSYM(41); 'GO TO' NWITEM; 'COMMENT' >= ;
0C218        'END' SPECSYM;
0C219
0C219       'COMMENT'  KWORDS IS THE PROCEDURE WHICH DOES APPROPRIATE ACTION
0C219          FOR RESERVED WORDS. IT BRANCHES ON CODE1 WHICH IS THE TABLE
0C219          ENTRY FOR THE RESERVED WORD IN ITEM . NORMALLY CODE8 IS EXECUTED
0C219          WHICH QUOTES THE RESERVED WORD;
0C219       'PROCEDURE' KWORDS(LENGTH,ITEM,CODE1);
0C220          'VALUE' CODE1; 'INTEGER' CODE1,LENGTH; 'INTEGER' 'ARRAY' ITEM;
0C223       'BEGIN'
0C223          'SWITCH' CASE:= CASE1,CASE2,CASE3,CASE4,CASE5,CASE6,CASE7,CASE8,
0C223             CASE9,CASE10,CASE11,CASE12,CASE13,CASE14,CASE15;
0C224          'INTEGER' I;
0C225          'GO TO' CASE(/MPC(/CODE1/)  /);
0C226       'COMMENT' CASE1 IS GO TO CASE WHERE AFTER GO IS RECOGNIZED A
0C226          SCAN FINDS THE NEXT C AND EMITS 'GOTO'. ***************;
0C226       CASE1:  ITEM(/1/):=QUOMRK; ITEM(/2/):=0; ITEM(/3/):=0;
0C229          OUTIDEN(3,ITEM);  SCANTILL(0);  QUTSYM(0); QUTSYM(QUOMRK);
0C233          'GO TO' NWITEM;
0C234       CASE2:  QUTSYM(52); 'GO TO' NWITEM ; 'COMMENT'  OR BECOMES |;
0C236       CASE3:  QUTSYM(45); 'GO TO' NWITEM ;    'COMMENT' AND BECOMES &;
0C238       CASE4: 'GO TO' CASE8;   'COMMENT' CURRENT HANDLING OF END***;
0C239       CASE5:  QUTSYM(44); 'GO TO' NWITEM ;    'COMMENT' NOT BECOMES ¬;
0C241       CASE6:      'FOR' I:=LENGTH 'STEP' -1 'UNTIL' 1 'DO'
0C241          ITEM(/I+1/):=ITEM(/I/);  LENGTH:=LENGTH+2;
0C243          ITEM(/1/):=ITEM(/LENGTH/):=QUOMRK; OUTIDEN(LENGTH,ITEM);
0C245       QUTSYM(BLANK);  SCANTILL(50);
0C247       'COMMENT' 50 IS A ;     PREV:='FALSE'; 'GO TO' NWITEM;
0C249       CASE7:  OUTDELETE('('THE FOLLOWING HAS BEEN DELETED FROM YOUR PROGRA
0C249    M')',LENGTH,ITEM); SCANTILL(50);  'GO TO' NWITEM;  'COMMENT' DELETION OF
0C252          LABEL , DEFINE AND FORWARD DECLARATIONS.***********;
0C252       CASE8:  'FOR' I:=LENGTH 'STEP' -1 'UNTIL' 1 'DO'
0C252          ITEM(/I+1/):=ITEM(/I/);
0C253          LENGTH:=LENGTH+2;  ITEM(/1/):=ITEM(/LENGTH/):=QUOMRK;
0C255          'GO TO' KWORDS;          'COMMENT' STANDARD CASE QUOTE RESERVED
0C256             WORDS ;
0C256       CASE9:  QUTSYM(QUOMRK); QUTSYM(40); QUTSYM(QUOMRK); 'GO TO' NWITEM;
0C260          'COMMENT'   DIV BECOMES '/'   ;
0C260       CASE10: ITEM(/3/):=0; ITEM(/4/):=19;  ITEM(/5/):=V; LENGTH:=5;
0C264          'GO TO' CASE8;  'COMMENT' EQV BECOMES 'EQUIV' ;
0C265       CASE11:  LENGTH:=4; ITEM(/4/):=22; 'GO TO' CASE8;
0C268          'COMMENT'  IMP BECOMES 'IMPL'  ;
0C268       CASE12:     'GO TO' TERMINATE;
0C269       CASE13:  OUTDELETE('(' OWN DECLARATION PLEASE  PUT IN OUTER BLOCK')'
0C269             ,3,ITEM);       'GO TO' NWITEM;  'COMMENT'  OWN DELETION;
0C271       CASE14:  OUTDELETE('(' ALPHA DECLARATION IS REPLACED BY REAL')',
0C271          LENGTH,ITEM);      ITEM(/1/):=26; ITEM(/2/):=15; ITEM(/3/):=11;
0C275          ITEM(/4/):=22;  LENGTH:=4; 'GO TO' CASE8;
```

```
SC      SOURCE STATEMENT

0027C       CASE15:  OUTDELETE('('THE FOLLOWING HAS BEEN DELETED FROM YOUR PROGR
0027C   AM')',LENGTH,ITEM); SCANENDOFSTAT; 'GO TO' NXITEM; 'COMMENT' DELETION
0028C          OF STATEMENTS WHICH CAN NOT BE AUTOMATICALLY TRANSLATED. IO,
0028C          FILL,DOUBLE STATEMENTS ARE OF THIS TYPE.************  ;
0028C     KEYWORDS:
0028C       'END' KEYWORDS;
0028C
0028C
0028C       'COMMENT' NXTITEM GETS NEXT ITEM TO TRANSLATE. THEY MAY BE
0028C          OF FOUR TYPES .  RESERVED WORDS  IDENTIFIERS  NUMBERS  SPECIAL
0028C          SYMBOLS ... IT PASSES THE ITEM AND ITS LENGTH AND A CODE
0028C          FOR THE ITEM TO THE TRANSLATE ROUTINE.   *********;
0028C       'PROCEDURE' NXTITEM(LENGTH,ITEM,CODE);
0028C          'INTEGER' LENGTH,CODE, 'INTEGER' 'ARRAY' ITEM;
0028C       'BEGIN' 'COMMENT' GETS NEXT SYNTACTICAL ITEM;
0028C          'IF' PREV 'THEN' SYMB:=NXTSYM;
0028C          'IF' SYMB=BLANK 'THEN'
0028C        'BEGIN' LENGTH:=1; CODE:=4;      ITEM(/1/):=SYMB; PREV:='TRUE' 'END'
0028C          'ELSE' 'IF'  SYMB<LETTERO  'THEN'
0028C             'BEGIN'
0028C                'IF' SYMB>DIGIT 'THEN'
0028C                'BEGIN'  ITEM(/1/):=SYMB;  INIDEN(LENGTH,ITEM);
0029C                   CODE:= 'IF' KEYWORD(ITEM,LENGTH) 'THEN' CUOTIT 'ELSE'
0029C                      IDEN;
0029C                'END'
0029C             'ELSE' 'BEGIN' ITEM(/1/):=SYMB; INNUME(LENGTH,ITEM); CODE:=
0029C                NUME; 'END'
0029C          'END'
0029C             'ELSE'  'BEGIN' LENGTH:=1; CODE:=SYMCOD; ITEM(/1/):=SYMB;
0029C                PREV:='TRUE';
0029C             'END';
0030C       'END'  NXTITEM;
0030C
0030C       'COMMENT' TRANSLATE TRANSLATES ITEM TO OUTPUT FORM.  ********    ;
0030C       'PROCEDURE' TRANSLATE(LENGTH,ITEM,CODE);
0030C          'VALUE' CODE;          'INTEGER' LENGTH,CODE;
0030C          'INTEGER' 'ARRAY' ITEM;
0030C       'BEGIN'
0030C          'SWITCH'  CASE:= CASE1,CASE2,CASE3,CASE4;
0030C         'GOTO' CASE(/CODE/);
0030C     CASE1:  KWORDS(LENGTH,ITEM,CODE1);
0030C     CASE2:  OUTIDEN(LENGTH,ITEM);  'GOTO' NXITEM;
0031C     CASE3:  OUTNUM(LENGTH,ITEM), 'GO TO' NXITEM;
0031C     CASE4:  SPECSYM(SYMB);
0031C       'END' TRANSLATE;
0031C
0031C       'COMMENT'  S T A R T   O F   M A I N    P R O G R A M   ********  ;
0031C       'COMMENT'  INITIALIZE DATA SETS***************************** ;
```

```
SC      SOURCE STATEMENT

00314       SYSACT(1,12,1);
00315       SYSACT(2,6,60); SYSACT(2,12,1);
00317       SYSACT(3,6,60); SYSACT(3,12,1),   SYSACT(1,15,1);
00320       'COMMENT' INITIALIZE TABLES AND SPECIAL VALUES ********************;
00320       MAXLEN:=10;  PREV:='TRUE';  CP:=73; COR:=1;  QUOTIT:=1; IDEN:=2;
00326       BLANK:=0; COMMRK:=49;   LETORDIG:=37; DIGIT:=10; NUMB:=3; SYMCOD:=4;
00332       CRLCNT:=1;  NUMCRD:=1;   SCRSCNT:=1;
00335       'FOR' I:=0 'STEP' 1 'UNTIL' 500 'DO' MPO(/I/):=0;
00336       'FOR' I:=1 'STEP' 1 'UNTIL' 5 'DO'
00336       'FOR' J:=1 'STEP' 1 'UNTIL' 80 'DO' CRD(/1,J/):=0;
00337       MPO(/3/):=1;  MPO(/5/):=2;  MPO(/9/):=3;  MPO(/15/):=4;
00341       MPO(/30/):=5;  MPO(/138/):=6;   MPO(/12/):=9;  MPO(/18/):=10;
00345       MPO(/120/):=7;  MPO(/24/):=11;  MPO(/192/):=7;
00348       MPO(/39/):=13;  MPO(/61/):=15;  MPO(/57/):=15; MPO(/45/):=15;
00352       MPO(/125/):=15; MPO(/231/):=7, MPO(/237/):=15; MPO(/243/):=7;
00356       MPO(/168/):=12;   MPO(/249/):=7,  MPO(/65/):=14;
00359       'FOR' I:=0 'STEP' 1 'UNTIL' 65 'DO'  MAP(/I/):=8;
00360          MAP(/44/):= 1;  MAP(/49/):=5,                  MAP(/51/):=4;
00363       MAP(/52/):=5;   MAP(/53/):=6,  MAP(/54/):=7;  MAP(/55/):=9;
00367       MAP(/56/):=2;   MAP(/60/):=10,
00369       'COMMENT' RWD CONTAINS TYPE 1 RESERVED WORDS FROM 65500  ;
00369        A:=11;  B:=12; C:=13; D:=14, E:=15; F:=16; G:=17; H:=18; I:=19;
00376        J:=20; K:=21; L:=22; M:=23, N:=24; O:=25; P:=26; Q:=27; R:=28;
00377        S:=29; T:=30; U:=31; V:=32; W:=33; X:=34; Y:=35; Z:=36;
00395       'COMMENT'  RW ENTRIES LENGTH 1 OR 3   ;
00395       TABLE(/1/):=TABLE(/6/):=ENTAB(/1/):=ENTAB(/8/):=3;
00396       TABLE(/2/):=1;
00397       RWD(/  1/):=D;  RWD(/  2/):=O;  RWD(/  3/):=G;  RWD(/  4/):=O;
00401       RWD(/  5/):=O;  RWD(/  6/):=K;  RWD(/  7/):=I;  RWD(/  8/):=F;
00405       ENTAB(/2/):=6;
00406       TABLE(/3/):=9;
00407       RWD(/  9/):=A;  RWD(/ 10/):=N,  RWD(/ 11/):=L;  RWD(/ 12/):=D;
00411       RWD(/ 13/):=I;  RWD(/ 14/):=V;  RWD(/ 15/):=E;  RWD(/ 16/):=N;
00415       RWD(/ 17/):=D;  RWD(/ 18/):=E;  RWD(/ 19/):=G;  RWD(/ 20/):=V;
00419       RWD(/ 21/):=F;  RWD(/ 22/):=O;  RWD(/ 23/):=R;  RWD(/ 24/):=I;
00423       RWD(/ 25/):=M;  RWD(/ 26/):=P,  RWD(/ 27/):=M;  RWD(/ 28/):=O;
00427       RWD(/ 29/):=L;  RWD(/ 30/):=N,  RWD(/ 31/):=L;  RWD(/ 32/):=I;
00431       RWD(/ 33/):=L;  RWD(/ 34/):=O,  RWD(/ 35/):=I;  RWD(/ 36/):=O;
00435       RWD(/ 37/):=O;  RWD(/ 38/):=I;  RWD(/ 39/):=L;  RWD(/ 40/):=A;
00439       RWD(/ 41/):=N;  RWD(/ 42/):=Z;  RWD(/ 43/):=I;  RWD(/ 44/):=P;
00443       ENTAB(/3/):=44;
00444       TABLE(/4/):=45;
00445       RWD(/ 45/):=O;  RWD(/ 46/):=O;  RWD(/ 47/):=M;  RWD(/ 48/):=P;
00449       RWD(/ 49/):=O;  RWD(/ 50/):=L,  RWD(/ 51/):=S;  RWD(/ 52/):=L;
00453       RWD(/ 53/):=F;  RWD(/ 54/):=I,  RWD(/ 55/):=L;  RWD(/ 56/):=E;
00457       RWD(/ 57/):=F;  RWD(/ 58/):=I,  RWD(/ 59/):=L;  RWD(/ 60/):=L;
00461       RWD(/ 61/):=R;  RWD(/ 62/):=L,  RWD(/ 63/):=A;  RWD(/ 64/):=O;
00465       RWD(/ 65/):=R;  RWD(/ 66/):=L,  RWD(/ 67/):=A;  RWD(/ 68/):=L;
```

```
SC      SOURCE STATEMENT

0C465       RWL(/ 69/):=S;  RWD(/ 70/):=I;  RWD(/ 71/):=E;  RWD(/ 72/):=P;
0C473       RWL(/ 73/):=I;  RWD(/ 74/):=M,  RWD(/ 75/):=L;  RWL(/ 76/):=R;
0C477       RWL(/ 77/):=I;  RWD(/ 78/):=R;  RWD(/ 79/):=L;  RWL(/ 80/):=E;
0C4E1       RWL(/ 81/):=R;  RWD(/ 82/):=I;  RWD(/ 83/):=T;  RWL(/ 84/):=F;
0C4E5       ENTAB(/4/):=84;
0C4E6       TABLE(/5/):=85;
0C4E7       RWL(/ 85/):=A;  RWD(/ 86/):=L,  RWD(/ 87/):=P;  RWL(/ 88/):=F;
0C4F1       RWL(/ 89/):=M;  RWD(/ 90/):=A,  RWD(/ 91/):=R;  RWL(/ 92/):=R;
0C4F9       RWL(/ 93/):=M;  RWD(/ 94/):=Y,  RWD(/ 95/):=R;  RWL(/ 96/):=E;
0C499       RWL(/ 97/):=G;  RWD(/ 98/):=I,  RWD(/ 99/):=R;  RWD(/100/):=E;
0C5C3       RWL(/101/):=A;  RWD(/102/):=L;  RWD(/103/):=S;  RWD(/104/):=E;
0C5C7       RWL(/105/):=L;  RWD(/106/):=N;  RWD(/107/):=T;  RWL(/108/):=I;
0C511       RWL(/109/):=L;  RWD(/110/):=V,  RWD(/111/):=A;  RWD(/112/):=L;
0C519       RWL(/113/):=L;  RWD(/114/):=E,  RWD(/115/):=R;  RWL(/116/):=F;
0C51D       RWL(/117/):=I;  RWD(/118/):=L,  RWD(/119/):=E;  RWD(/120/):=L;
0C523       RWL(/121/):=A;  RWD(/122/):=S,  RWD(/123/):=E;  RWL(/124/):=L;
0C527       RWL(/125/):=R;  RWD(/126/):=R,  RWD(/127/):=I;  RWD(/128/):=T;
0C531       RWL(/129/):=E;
0C532       ENTAB(/5/):=129;
0C533       TABLE(/6/):=225;
0C534       RWL(/225/):=S;  RWD(/226/):=R,  RWD(/227/):=I;  RWL(/228/):=T;
0C53E       RWL(/229/):=L;  RWD(/230/):=N;
0C540       RWL(/231/):=F;  RWD(/232/):=O,  RWD(/233/):=R;  RWD(/234/):=M;
0C544       RWL(/235/):=A;  RWD(/236/):=T;
0C54E       RWL(/237/):=L;  RWD(/238/):=O,  RWD(/239/):=L;  RWD(/240/):=B;
0C550       RWL(/241/):=L;  RWD(/242/):=E,
0C55A       RWL(/243/):=O;  RWD(/244/):=L;  RWD(/245/):=F;  RWL(/246/):=I;
0C55E       RWL(/247/):=N;  RWD(/248/):=E;
0C55E       RWL(/249/):=S;  RWD(/250/):=I;  RWD(/251/):=R;  RWD(/252/):=E;
0C5E2       RWL(/253/):=A;  RWD(/254/):=M;
0C5E4       ENTAB(/6/):=254;
0C5E5       TABLE(/7/):=131;
0C5E6       RWL(/131/):=J;  RWD(/132/):=O,  RWD(/133/):=L;  RWL(/134/):=L;
0C5F0       RWL(/135/):=L;  RWD(/136/):=A;  RWD(/137/):=N;  RWD(/138/):=C;
0C5F4       RWL(/139/):=L;  RWD(/140/):=M;  RWD(/141/):=M;  RWL(/142/):=E;
0C5FE       RWL(/143/):=N;  RWD(/144/):=I,  RWD(/145/):=I;  RWD(/146/):=N;
0C582       RWL(/147/):=I;  RWD(/148/):=L,  RWD(/149/):=G;  RWL(/150/):=E;
0C586       RWL(/151/):=N;  RWD(/152/):=F,  RWD(/153/):=L;  RWD(/154/):=R;
0C590       RWL(/155/):=A;  RWD(/156/):=A,  RWD(/157/):=R;  RWD(/158/):=D;
0C594       ENTAB(/7/):=156;
0C595       TABLE(/8/):=159;
0C596       RWL(/159/):=P;  RWD(/160/):=R;  RWD(/161/):=E;  RWD(/162/):=C;
0C600       RWL(/163/):=L;  RWD(/164/):=O,  RWD(/165/):=L;  RWD(/166/):=R;
0C604       RWL(/167/):=L;
0C60B       RWL(/168/):=F;  RWD(/169/):=I;  RWD(/170/):=N;  RWL(/171/):=L;
0C609       RWL(/172/):=F;  RWD(/173/):=I;  RWD(/174/):=R;  RWL(/175/):=A;
0C613       RWL(/176/):=N;
0C614       ENTAB(/8/):=176;
```

```
 SC      SOURCE STATEMENT

0C615         'COMMENT' END OF SYMBOL TABLE FOR RESERVED WORDS**********;
0C615         'COMMENT' THIS RESERVED WORD TABLE SET UP IS BASED ON AN IDEA OF SG;
0C615       NWITEM:
0C615         NXTITEM(LENGTH,ITEM,CODE);
0C616         TRANSLATE(LENGTH,ITEM,CODE);
0C617          'GO TO' NWITEM;
0C618       TERMINATE:
0C618         'END'
```