FORTRAN IV TO PL/1 TRANSLATOR

AUTHOR: Lanse M. Leach

Direct Inquiries To: Lanse M. Leach
Computation Center
Stanford University
Stanford, California 94305

PART I

DESCRIPTION AND GUIDE TO THE TRAMSLATOR PROGRAM

This section contains a detailed description of the translator program with enough facts to enable a programmer to expand this translator to handle additional FORTRAN statements or to modify the translator to accept the FORTRAN of a particular computer installation instead of the ASA Standard FORTRAN.

The translator program, written in PL/1and compiled under the IBM System/360 F-Level PL/1 Compiler, consists of a main program of four executable statements and fifty-one internal procedures. The main program transfers control to the procedure PROGRAM for each FORTRAN main or sub-program in the input stream, Execution terminates when an end of file is encountered in the input stream.

Procedure PROGRAM performs the initialization of the symbol table and various counters used by the translator and calls procedure STATEMENT for each FORTRAN statement until the FORTRAN END statement is encountered. The procedure OUTPUT is called which produces PL/1 declarations from the symbol table information and transfers PL/1 statements from two temporary files or data-sets to the printer and punch or directly to a data-set for input to the PL/1 Compiler.

The text scanning procedure SCAN makes extensive use of PL/1 string operations to return the varying length character string, NEXT, which is composed of either a string of letters, a string of digits, or a special character. Thus to recognize the FORTRAN floating point number 5.58E-16 six calls to SCAN are required. Also it is the responsibility of procedure FORTIDEN to build valid FORTRAN identifiers from repeated calls to SCAN.

SCAN has a one bit argument which controls whether or not to start scanning a new FORTRAN statement or to continue with the current input string, LINE. Other functions performed by SCAN are label processing and comment processing. SCAN determines if a statement number exists on a new statement and if so calls procedure LABEL. If a "C" occurs in column one then procedure COMMENT is called directly from SCAN. SCAN also checks the next card in the input stream for a continuation mark, If the card is a continuation, columns seven to seventy-two are concatenated onto LINE and the next card in the input stream is checked. Thus the reading of card images from the input stream is always one card ahead of the actual processing of FORTRAN statements. Because of this and in order that the translator can have any number of FORTRAN main or subprograms in the input stream, a blank card is required after each END statement in the input stream,

FORTRAN comments have blanks chopped off each end and if a comment card contains all blanks with the exception of the "C" in column one, a blank card is inserted in the PL/1 program to improve appearance. The following is an example of comment processing:

FORTRAN COMMENT

```
C     THIS IS A COMMENT CARD ..................
```

TRANSLATOR PRODUCED PL/1 COMMENT

```
/* THIS IS A COMMENT CARD ..............  */
```

A symbol table is constructed from the information provided in the FORTRAN type statements and consists of four varying length character string arrays as follows:

SYMBOL	NAME OF THE FORTRAN VARIABLE
SYMDIM	DIMENSIONING INFORMATION
SYMTYPE	TYPE INFORMATION
SYMCOM	EXTERNAL AND INITIAL ATTRIBUTE INFORMATION

For each FORTRAN type statement encountered the symbol table is checked to see if the FORTRAN variable is already in the symbol table and if so the symbol is updated otherwise the new variable name is added to the end of the symbol table and the length counter is increased by one.

Translator input/output is done using PL/1 stream input/output statements. Only when reading FORTRAN card images is a formatted input/output statement used. At all other times either a PUT LIST or a GET LIST input/output statement is used. The following are the five input/output files used by the translator.

SYSIN	INPUT	Fortran source card images
DECLIST	INPUT/ OUTPUT	Scratch file for PL/1 declarations and the initial procedure declaration. On the IBM 360/67 this scratch file was on a 2311 disk pack.
PRGLIST	INPUT/ OUTPUT	Scratch file for PL/1 statements other than declarations. On the IBM 360/67 this scratch file was on a 2311 disk pack
SYSPRINT	OUTPUT	Printed output usually consisting of input

		FORTRAN programs, error messages, and the translator produced PL/1 program.
PUNLIST	OUTPUT	Output of the translator produced PL/1 program Punch, data-set for input to the PL/1 Compiler, or dummy if no supplementary output is desired

In order to produce readable PL/1 output, a tab consisting of a variable number of blanks is concatenated on the front of each output string. Initially the tab is of length one and is increased by three blanks each time a DO group or procedure is entered and decreased by three blanks upon exiting a DO group or procedure.

In order to modify the translator to accept additional FORTRAN statements the following two steps are required.

(1) Write an internal procedure to translate the additional statement. To output the translated statement a call to procedure DISK with the output character string as argument is required.

(2) Insert an IF statement in procedure PROCEDURE to produce a call to the added internal procedure when the added statement is encountered.

PART II

LIMITATION AND PROGRAMMER INTERVENTION REQUIRED

The following areas require the user of the translator to modify the translator-produced PL/i in order to obtain a correctly running PL/I program:

(1) DATA, EQUIVALENCE, and BACKSPACE statements are not translated and thus require hand translation. For the EQUIVALENCE statement the DEFINED attribute in the DECLARE statement can be used to obtain a correct translation. For the DATA statement the INITIAL attribute in the DECLARE statement can be used. In PL/1 there is no direct translation for the BACKSPACE statement. Thus a FORTRAN program using the BACKSPACE statement would require modification.

(2) Hollerith strings in FORTRAN FORMAT statements can not be placed in PL/1 FORMAT statements. Thus hollerith strings must be converted to character strings and placed in the input or output statement variable lists.

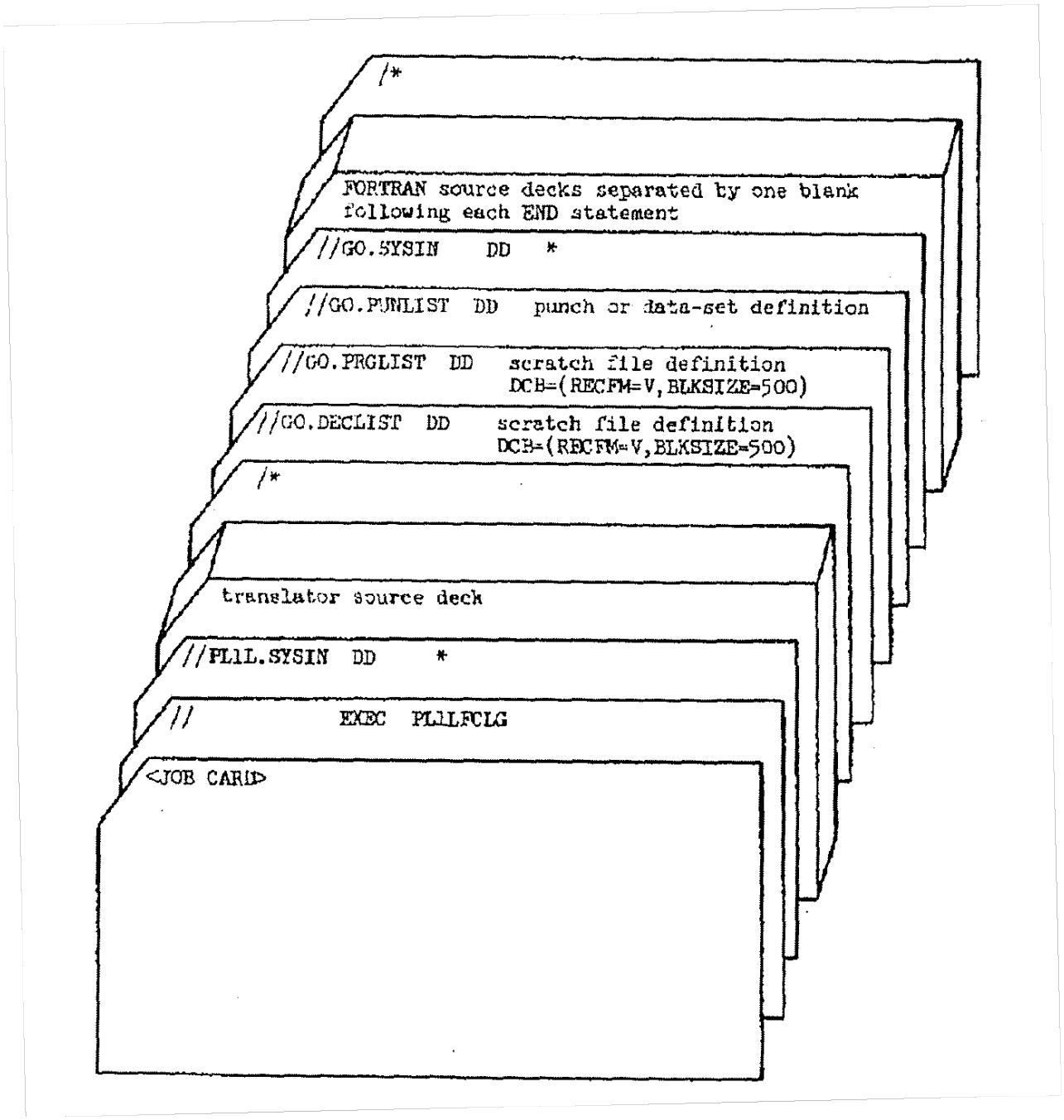
(3) FORTRAN binary input and output statements (unformatted) are translated into GET and PUT LIST statements which are unformatted but not binary in the same sense as in FORTRAN. If this is not acceptable, hand translation to RECORD input and output statements is required.

(4) Input FORMAT lists for card image input must be of length 80 characters (columns) in order to use STREAM input and get the same effect as in FORTRAN. Example one requires this modification in order to execute correctly.

(5) Variables which were in COMMON in the FORTRAN program and were declared external in the PL/1 version must be checked to be sure that different names for the same location are not used in a separate subprogram.

(6) Blanks are significant and FORTRAN is assumed to have reserved words. Any violation of this simplification must be hand translated.

SAMPLE JOB CONTROL LANGUAGE FOR EXECUTION OF THE TRANSLATOR



NOTE: PL1LFCLG is the standard IBM cataloged procedure for PL/1 compile, link, and go.

EXAMPLE 1

FORTRAN SOURCE PROGRAM

```
C     SOLUTION OF SIMULTANEOUS EQUATIONS BY GAUSSIAN ELIMINATION
C
100   FORMAT (I5)
101   FORMAT (8F10.2)
102   FORMAT( I5,F20.2)
      DIMENSION A(50,50),Y(50),X(50)
      READ (5,100) N
      READ (5,101) ((A(I,J),J=1,N),I=1,N)
      READ (5,101) (Y(I),I=1,N)
      M=N-1
      DO 10 I=1,M
      L=I+1
      DO 10 J=L,N
      IF (A(J,I)) 6,10,6
6     DO 8 K=L,N
8     A(J,K)=A(J,K)-A(I,K)*A(J,I)/A(I,I)
      Y(J)=Y(J)-Y(I)*A(J,I)/A(I,I)
10    CONTINUE
      X(N)=Y(N)/A(N,N)
      WRITE (6,102) N,X(N)
      DO 30 I=1,M
      K=N-I
      L=K+1
      DO 20 J=L,N
20    Y(K)=Y(K)-X(J)*A(K,J)
      X(K)=Y(K)/A(K,K)
30    WRITE (6,102) K,X(K)
      RETURN
      END
```

EXAMPLE 1 (CONTINUED)

PL/1 VERSION OF FORTRAN PROGRAM

```
/* FORTRAN PROGRAM TRANSLATED TO PL/1 */
FORT: PROCEDURE OPTIONS (MAIN);
   DECLARE A(50,50);
   DECLARE Y(50);
   DECLARE X(50);
   /* SOLUTION OF SIMULTANEOUS EQUATIONS BY GAUSSIAN ELIMINATION */

   #100: FORMAT (F(5));
   #101: FORMAT (8F(10,2));
   #102: FORMAT (F(5),F(20,2));
   GET FILE (SYSIN)EDIT(N)(R(#100));
   GET FILE (SYSIN)EDIT(((A(I,J) DO J=1 TO N BY 1) DO I=1 TO N BY
    1))(R(#101));
   GET FILE (SYSIN)EDIT((Y(I) DO I=1 TO N BY 1))(R(#101));
   M= N - 1;
   DO I=1 TO M BY 1;
      L= I + 1;
      DO J=L TO N BY 1;
         IF (A(J,I)) > 0 THEN GO TO #6;
         ELSE IF (A(J,I)) = 0 THEN GO TO #10;
         ELSE GO TO #6;
         #6: DO K=L TO N BY 1;
            #8: A(J,K)= A(J,K) - A(I,K)*A(J,I)/A(I,I);
            END;
         Y(J)= Y(J) - Y(I)*A(J,I)/A(I,I);
         #10:
         END;
      END;
   X(N)= Y(N)/A(N,N);
   PUT FILE (SYSPRINT)EDIT(N,X(N))(R(#102));
   DO I=1 TO M BY 1;
      K= N - I;
      L= K + 1;
      DO J=L TO N BY 1;
         #20: Y(K)= Y(K) - X(J)*A(K,J);
         END;
      X(K)= Y(K)/A(K,K);
      #30: PUT FILE (SYSPRINT)EDIT(K,X(K))(R(#102));
      END;
   RETURN;
END;
```

EXAMPLE 2

FORTRAN SOURCE PROGRAM

```
      SUBROUTINE POLAR (X,Y,R,THETA)
C     CONVERT CARTESIAN TO POLAR COORDINATES
      R=SQRT(X*X+Y*Y)
      THETA=ATAN2(Y,X)
      RETURN
      END
```

PL/1 VERSION OF FORTRAN PROGRAM

```
 /* FORTRAN PROGRAM TRANSLATED TO PL/1 */
 POLAR:PROCEDURE(X,Y,R,THETA);
    /* CONVERT CARTESIAN TO POLAR COORDINATES */
    R= SQRT(X*X + Y*Y);
    THETA= ATAN(Y,X);
    RETURN;
 END;
```

EXAMPLE 3

FORTRAN SOURCE PROGRAM

```
      FUNCTION HELP (A,B,C)
      IF (A.GT.B) GO TO 500
      HELP=B*C
      RETURN
500   HELP=A*C
      RETURN
      END
```

PL/1 VERSION OF FORTRAN PROGRAM

```
 /* FORTRAN PROGRAM TRANSLATED TO PL/1 */
 HELP: PROCEDURE (A,B,C);
    IF (A > B) THEN
    GO TO #500;
    HELP#= B*C;
    RETURN (HELP#);
    #500: HELP#= A*C;
    RETURN (HELP#);
 END;
```

The remainder of the original writeup consisted of the complete listing of the translator program. The listing is omitted here for brevity.

The original program source is available in the file f4_to_pli.original.

ADDENDUM

The following changes were made to produce the current version of the program:

(1) The source was reformatted using the Cornell PL/I reformatter with the default settings.

(2) The declarations of internal procedures EXPRESSION thru PRIMARY are not allowed by current compilers and were removed.

(3) Procedures EXPRESSION thru NUMBER needed the RETURNS attribute, which was apparently not required by the compiler originally used [OS PL/I(F) version 2],

(4) In procedure READWRITE, the lines IF NEXT='6' THEN UNIT='SYSPRINT'; and ELSE IF NEXT='5' THEN UNIT='SYSIN'; were interchanged to match what is shown on the listing, apparently the cards were shuffled as they were read in.

At this point the program was compiled using IBM's "PL/I for MVS and VM" compiler, version 1.1, and executed successfully.

(5) The modified source was downloaded to a PC and compiled for OS/2 using Iron Spring PL/I 0.8c.

(6) The code contained various dependencies on the EBCDIC character encoding, such as checks for a numeric character using "IF <character> >='0'". The EBCDIC collating sequence has all special characters low followed by alphabetic characters, with numbers high. These dependencies were fixed, but should be replaced by calls to procedures that identify alphabetics, numerics, and specials, similar to C's "isalpha", etc.

(7) SYSIN was changed to a RECORD file to read one line at a time and move it to the fixed 80-character 'card-image' record expected by the program.

The following need to be done:

(8) The translated PL/I is all upper-case, as required by the original target compiler, This code will compile, but the appearance could be improved by use of mixed-case.

(9) The generated PL/I contains some idiosyncrasies of the PL/I(F) compiler.

(10) IBM mainframe programs such as PL/I(F) use Job Control Language (JCL), as shown by the

example, to associate files in the program with “data-sets” on external media. Non mainframe compilers typically use the TITLE option on the OPEN statement. The names of the input and output files should be taken from the command-line and used as the TITLEs.

(11) FORTRAN IV as translated by this program is apparently now somewhat dated. The translator needs to be modified to accept a more current level of FORTRAN.

This program was originally released as SHARE program number 360D-12.2.002 dated August, 1967.

The program writeup, except for this addendum, has been formatted as closely as possible to match the original, which was judged non-OCRable. It is complete and unchanged, with the omission of the ten-page compile listing of the translator program. A copy of the original writeup can be found at http://cbttape.org/ftp/SPLA/SPLA_FULLDOC_PDF.zip as file 122022.pdf. The original author has no responsibility for the current version of the translator.

Peter Flass
<Peter_Flass@Yahoo.com>
September, 2009.