

AD-A031 460

GENERAL RESEARCH CORP SANTA BARBARA CALIF  
STRUCTURED PROGRAMMING TRANSLATORS. VOLUME IV. STRUCTRAN-2 USER--ETC(U)  
AUG 76 R A MELTON

F/G 9/2

F30602-75-C-0245

RADC-TR-76-253-VOL-4

NL

UNCLASSIFIED

1 OF 1  
AD  
A031460



END  
DATE  
FILMED  
12-76

AD A031460

RADC-TR-76-253, Vol IV (of five)  
Final Technical Report  
August 1976



STRUCTURED PROGRAMMING TRANSLATORS  
STRUCTRAN-2 User's Manual

General Research Corporation

**COPY AVAILABLE TO DDC DOES NOT  
PERMIT FULLY LEGIBLE PRODUCTION**

Approved for public release;  
distribution unlimited.

DDC  
RECEIVED  
NOV 2 1976  
RECEIVED  
D

ROME AIR DEVELOPMENT CENTER  
AIR FORCE SYSTEMS COMMAND  
GRIFFISS AIR FORCE BASE, NEW YORK 13441

This report consists of the following volumes:

- I - Final Report
- II - STRUCTRAN-1 User's Manual
- III - STRUCTRAN-1 System Design and Implementation Manual
- IV - STRUCTRAN-2 User's Manual
- V - STRUCTRAN-2 System Design and Implementation Manual

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

This report has been reviewed and approved for publication.

APPROVED:

*Donald L. Mark*

DONALD L. MARK  
Project Engineer

APPROVED:

*Robert D. Krutz*

ROBERT D. KRUTZ, Col, USAF  
Chief, Information Sciences Division

FOR THE COMMANDER:

*John P. Huss*

JOHN P. HUSS  
Acting Chief, Plans Office

Do not return this copy. Retain or destroy.

*MISSION*  
*of*  
*Rome Air Development Center*

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C<sup>3</sup>) activities, and in the C<sup>3</sup> areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER RADC-TR-76-253 - Vol. <u>4</u>	2. GOVT ACCESSION NO.	3. PRICE STATE CATALOG NUMBER
4. TITLE (and Subtitle) STRUCTURED PROGRAMMING TRANSLATORS, Volume IV. STRUCTRAN-2 User's Manual.	5. DATE OF REPORT & PERIOD COVERED Final Technical Report, May 75 - Jan 76.	6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR R. A. Melton	8. CONTRACT OR GRANT NUMBER(s) F30602-75-C-0245	
9. PERFORMING ORGANIZATION NAME AND ADDRESS General Research Corporation P. O. Box 3587 Santa Barbara CA 93105	10. PROGRAM ELEMENT PROJECT, TASK AREA & WORK UNIT NUMBERS 32010314	
11. CONTROLLING OFFICE NAME AND ADDRESS Rome Air Development Center (ISIS) Griffiss AFB NY 13441	12. REPORT DATE August 1976	13. NUMBER OF PAGES 16
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Same	15. SECURITY CLASS. (of this report) UNCLASSIFIED	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited	17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report) Same	
18. SUPPLEMENTARY NOTES RADC Project Engineer: Donald L. Mark (ISIS)	DMAAC Project Engineer Ms. Opal Power	
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Structured Programming Precompilers Translators Software Tools		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) STRUCTRAN-2 is a system which converts unstructured FORTRAN programs into structured form. The STRUCTRAN-2 User's Manual describes the use and operation of this system. STRUCTRAN-2 is compatible with the STRUCTRAN-1 precompiler which is a special purpose language preprocessor intended to augment standard FORTRAN and permit its use as the basis of a structured programming effort.		

DD FORM 1473 JAN 73 EDITION OF 1 NOV 65 IS OBSOLETE

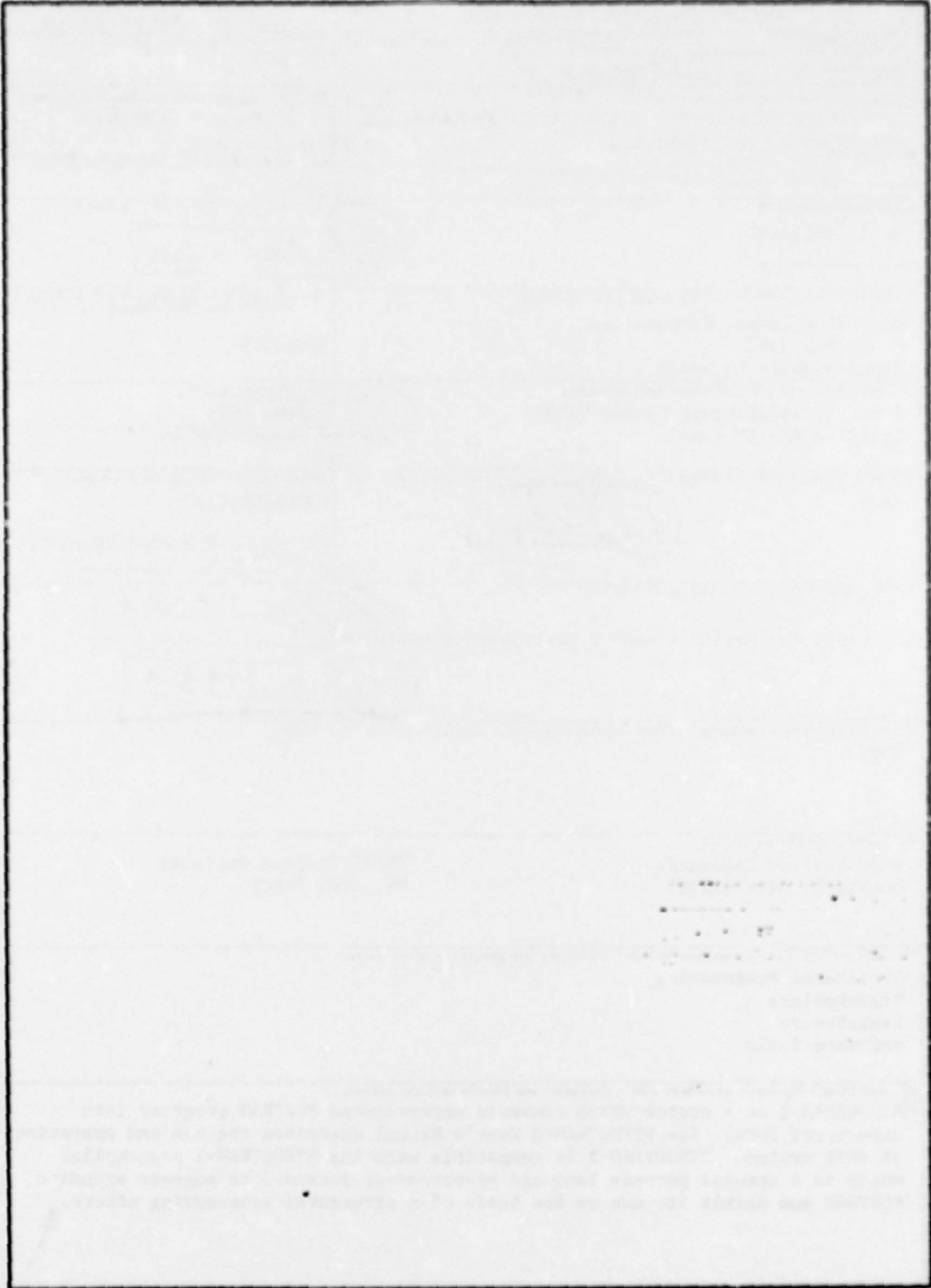
UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

402 754 LB

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)



UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

CONTENTS

<u>SECTION</u>		<u>PAGE</u>
1	INTRODUCTION	1
2	USING STRUCTRAN-2	2
	2.1 STRUCTRAN-2 Input	3
	2.2 STRUCTRAN-2 Output	3
	2.3 Default Parameters	3
3	STRUCTRAN-2 CONSTRAINTS	6
4	STRUCTRAN-2 PROGRAM OVERVIEW	7
	4.1 STRUCTRAN-2 Program Organization	7
	4.2 STRUCTRAN-2 Operation	8
	4.3 STRUCTRAN-2 Data Structures	9
APPENDIX A	USING STRUCTRAN-2 ON THE UNIVAC 1108	10
APPENDIX B	STRUCTRAN-2 SYSTEM DIAGNOSTICS (FOR USE WITH HIGH-LEVEL AUDIT)	11
APPENDIX C	HIGH-LEVEL AUDIT	14

ACCESSION No	
DTIC	Write Section <input checked="" type="checkbox"/>
DOC	Diff Section <input type="checkbox"/>
UNANNOUNCED	<input type="checkbox"/>
JUSTIFICATION	
BY	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	AVAIL. and/or SPECIAL
A	

DDC  
RECEIVED  
NOV 2 1976  
RECEIVED  
D

## ILLUSTRATIONS

<u>NO.</u>		<u>PAGE</u>
2.1	Use of STRUCTRAN-2	2
2.2	STRUCTRAN-2 Input Format	4
2.3	STRUCTRAN-2 Original FORTRAN Source Listing	4
2.4	STRUCTRAN-2 Output Format	4
4.1	STRUCTRAN-2 Control	8
C.1	Statement Descriptor Blocks for Module SUM	15
C.2	Symbol Table for Module SUM	15
C.3	DMATRAN Statements for Module SUM	16
C.4	DD-Paths for Module SUM	16

## 1 INTRODUCTION

The techniques of Structured Programming are finding increasing application in the computing community. Structured programs are, however, difficult to write in programming languages that do not have the statement types necessary to produce *GOTO-free* code. DMATRAN is a *GOTO-less* programming language that augments standard FORTRAN so that it can be used as a basis for structured programming. The STRUCTRAN-1 preprocessor translates DMATRAN into standard FORTRAN. It is compatible with the output of STRUCTRAN-2, a system which converts unstructured FORTRAN programs into structured form in DMATRAN. This "structurization" capability is useful when existing FORTRAN programs are to be maintained, modified, documented, or understood. The enhanced visibility of program structure provided by STRUCTRAN-2 translation aids in these tasks.

General use of STRUCTRAN-2 is discussed in Sec. 2. Appendix A describes the use of STRUCTRAN-2 on the UNIVAC 1108. STRUCTRAN-2 constraints are summarized in Sec. 3. Error messages which may occur during a STRUCTRAN-2 run are summarized in Appendix B, and the high-level audit which may be printed during STRUCTRAN-2 processing is described in Appendix C.

## 2 USING STRUCTRAN-2

STRUCTRAN-2 translates existing FORTRAN programs into structured DMATRAN programs, logically equivalent to the original FORTRAN programs. Structurization does not change the algorithm implemented by the original program, but reveals the structure of the algorithm so that it may be more readily understood. STRUCTRAN-2 is useful when existing FORTRAN programs are to be maintained, modified, documented, or studied. The DMATRAN program can be listed and executed by using the STRUCTRAN-1 preprocessor (refer to the STRUCTRAN-1 User's Manual).

STRUCTRAN-2 replaces FORTRAN control statements with the following four DMATRAN statement constructs:

- IF...THEN...ELSE...END IF--provides block structuring of conditionally executable sequences of statements
- DO WHILE...END WHILE--permits iteration of a code segment while a specified condition remains true
- DO UNTIL...END UNTIL--permits iteration until a specified condition becomes true
- BLOCK<name>...END BLOCK (and corresponding INVOKE<name> statement) --provide a facility for top-down programming and internal subroutines.

STRUCTRAN-2 file usage is shown in Fig. 2.1. The FORTRAN source program to be translated is read from the system card reader (unit number is installation-dependent). A listing of the original FORTRAN program (plus any FORTRAN

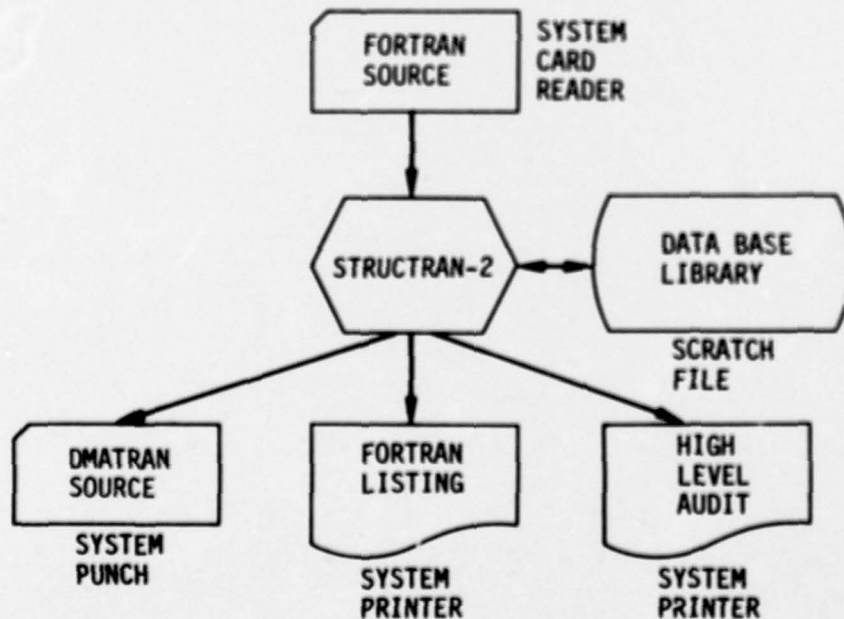


Figure 2.1. Use of STRUCTRAN-2

statements inserted by STRUCTRAN-2) is produced on the system printer. Infrequently, it is necessary to examine the high-level audit trail (Appendix C) produced by STRUCTRAN-2 during the structurization process. When this is the case, the scratch file associated with the audit must be copied to the system printer. The DMATRAN source code is written in card image form onto either the system punch or a permanent file. After the STRUCTRAN-2 run is complete, the STRUCTRAN-1 preprocessor can be used to list and execute the DMATRAN program produced.

## 2.1 STRUCTRAN-2 INPUT

The only input required by STRUCTRAN-2 is the FORTRAN source code to be structurized. It must be in card image form and should be compilable by the UNIVAC 1108 FORTRAN V compiler (or any compiler which implements a subset of FORTRAN V) with no errors. More than one routine may be structurized during a STRUCTRAN-2 run. No control cards between routines are required; STRUCTRAN-2 recognizes the end of each routine automatically. Additional constraints on the FORTRAN source code are summarized in Sec. 3. Figure 2.2 indicates the input format for STRUCTRAN-2.

## 2.2 STRUCTRAN-2 OUTPUT

STRUCTRAN-2 lists the FORTRAN source code, with any modifications which may have been made to it (Fig. 2.3). For example, a unique CONTINUE statement is added for any DO statement which was not associated with a unique CONTINUE in the original FORTRAN (line 11 of the example). The listing associates an internal STRUCTRAN-2 statement number with each statement. This statement number is useful in tracing the structurized code back to the original FORTRAN code. The DMATRAN source code is also output by STRUCTRAN-2 in card image format (Fig. 2.4), so that it can be readily processed by the STRUCTRAN-1 preprocessor. A sequence number in columns 73 through 80 of the DMATRAN source code relates each statement to the corresponding statement in the STRUCTRAN-2-produced FORTRAN listing. This sequence information is also displayed on the indented listing produced by STRUCTRAN-1.

It is often appropriate to examine the DMATRAN source code produced by STRUCTRAN-2 for the purpose of making simple but effective improvements. For example, if an original FORTRAN program contains duplicate segments of code, they will be highly visible in the structurized version because their structures will be identical. The BLOCK...END BLOCK and INVOKE statements can be conveniently used to eliminate these redundant segments of code by manually modifying the DMATRAN source code.

```

SUBROUTINE SUM(N,ARRAY,ISUM)
INTEGER ARRAY(10)
ISUM=0
IF(N.LE.0)GOTO 999
J=0
50 K=N-J
IF(K.GT.10)K=10
DO 100 I=1,K
ISUM=ISUM+ARRAY(I)
100 J=J+1
IF(J.LT.N)GOTO 50
999 RETURN
END

```

Figure 2.2. STRUCTRAN-2 Input Format

ORIGINAL FORTRAN SOURCE STATEMENTS FOR MODULE SUM .....

STMT	LABEL	STATEMENT TEXT....
1		SUBROUTINE SUM ( N , ARRAY , ISUM )
2		INTEGER ARRAY ( 10 )
3		ISUM = 0
4		IF ( N .LE. 0 ) GOTO 999
5		J = 0
6	50	K = N - J
7		IF ( K .GT. 10 ) K = 10
8		DO 10100 I = 1 , K
9		ISUM = ISUM + ARRAY ( I )
10	100	J = J + 1
11	10100	CONTINUE
12		IF ( J .LT. N ) GOTO 50
13	999	RETURN
14		END

Figure 2 3. STRUCTRAN-2 Original FORTRAN Source Listing

SUBROUTINE SUM ( N , ARRAY , ISUM )	1
INTEGER ARRAY ( 10 )	2
ISUM = 0	3
IF ( N .GT. 0 ) THEN	4
J = 0	5
DO UNTIL ( J .GE. N )	12
K = N - J	6
IF ( K .GT. 10 ) K = 10	7
I = 1	0
DO UNTIL ( I .GT. K )	11
ISUM = ISUM + ARRAY ( I )	9
J = J + 1	10
I = I + 1	11
END UNTIL	0
END UNTIL	0
END IF	0
RETURN	13
END	14

Figure 2.4. STRUCTRAN-2 Output Format

### 2.3 DEFAULT PARAMETERS

No provision has been made for user modification of the default parameters. The values for each are determined during STRUCTRAN-2 installation. Default values currently used are listed below:

<u>Parameter</u>	<u>Default</u>
STRUCTRAN-2 input unit	5LINPUT
STRUCTRAN-2 listing unit	6LOUTPUT
STRUCTRAN-2 error diagnostics unit	6LOUTPUT
STRUCTRAN-2 output unit	6LLPUNCH
Include comments in structurized source code	YES
BLOCK generation threshold	
Number of duplicated statements	15
Number of statements	1000
Number of paths	1,000,000
Working storage size	10,000

### 3 STRUCTRAN-2 CONSTRAINTS

The following constraints should be kept in mind when using STRUCTRAN-2:

1. There must be at least one blank after every source-text FORTRAN keyword (e.g., REAL I or DO 100 I=1,10, not REALI or DO100).
2. Only UNIVAC 1108 FORTRAN V statement forms are acceptable.
3. Only the ANSI X3.9 form of the ASSIGNED GOTO statement is processed (e.g., all labels to which control can transfer must be indicated in the ASSIGNED GOTO statement).
4. At most 250 keywords, operators, delimiters, and symbols are permitted in any statement (except FORMAT statements and Hollerith strings).
5. DO-statement targets are automatically retargeted when necessary, each to a distinct CONTINUE statement.
6. No decision-to-decision path (DD-path) in the program may contain more than 100 successive assignment statements or unconditional I/O statements.
7. Iterations which have more than one entry may not be correctly structured. An example of a multiple-entry iteration follows:

```
      IF(P) GO TO 10
5     IF(Q) GO TO 20
      I=I+1
10    J=J+1
      GO TO 5
20    CONTINUE
```

The iteration in this example can be entered at label "5" or label "10". The complexity of individual multiple entry iterations determines the results of the structurization. Several outcomes are possible:

- (1) Structurized source code is okay (the simple example above will be correctly structured).
- (2) Structurized source code contains a DO WHILE statement that is always true (modify the structured source code).
- (3) Structurization is not completed and an error message describing the program graph is printed (use the program graph to identify the multiple-entry iteration and modify the original FORTRAN source code).

## 4 STRUCTRAN-2 PROGRAM OVERVIEW

### 4.1 STRUCTRAN-2 PROGRAM ORGANIZATION

STRUCTRAN-2 is organized into eight components, each of which consists of a hierarchy of related modules with specific processing capabilities. These components are used in a definite order to translate FORTRAN to DMATRAN. Table 4.1 shows the STRUCTRAN-2 components; detailed descriptions of their functions appear in Sec. 4 of the STRUCTRAN-2 System Design and Implementation Manual.

---

TABLE 4.1  
STRUCTRAN-2 COMPONENTS

<u>Component</u>	<u>Memory (Octal)</u>	<u>Function</u>
STRUC0	77K	Storage Manager, Data Base Services, Support Subroutines, and Control
STRUC1	3K	Initialization of Data Base
STRUC2	14K	Lexical Analysis
STRUC3	14K	FORTRAN Parsing
STRUC4	13K	Graph Identification
STRUC5	16K	Graph Analysis
STRUC6	20K	Formation of DMATRAN Program
STRUC7	5K	Print/Punch Services

---

The UNIVAC-1108 installation of STRUCTRAN-2 operates in an overlay mode. STRUC0 is the root of the overlay structure and causes each of the other components to be loaded and executed in the proper order. All routines that are required by two or more components are included in STRUC0. STRUC0 is always in core, with at most one other component. The approximate size of each component as overlaid on the UNIVAC 1108 is shown in Table 4.1.

The code for all STRUCTRAN-2 components was developed by GRC personnel, utilizing structured programming techniques throughout.

## 4.2 STRUCTRAN-2 OPERATION

Each STRUCTRAN-2 component is devoted to a unique task; communication between components is through a common data base called a data base library. The data base library is initially created by STRUC1, modified by STRUC2, STRUC3, STRUC4, and STRUC5, and utilized by STRUC6 and STRUC7. The sequence in which the components are executed is shown in Fig. 4.1, which uses DMATRAN to express a high-level, structured representation of STRUCTRAN-2's operation.

---

```
PROCEDURE STRUCTRAN2 CONTROL
  INVOKE ( STRUC1 = INITIALIZE DATA BASE )
  DO WHILE ( END OF THE CURRENT MODULE HAS NOT BEEN REACHED )
    . INVOKE ( STRUC2 = PERFORM LEXICAL SCAN OF MODULE TO IDENTIFY
1. .   KEYWORDS AND VARIABLES )
    . IF ( END OF THE CURRENT MODULE HAS NOT BEEN REACHED ) THEN
    . . INVOKE ( STRUC3 = PERFORM PARSE OF MODULE TO IDENTIFY STATEMENT
1. . .   TYPES )
    . . INVOKE ( STRUC7 = PRINT FORTRAN MODULE )
    . . INVOKE ( STRUC4 = IDENTIFY PROGRAM GRAPH )
    . . INVOKE ( STRUC5 = RECOGNIZE STRUCTURED FORMS IN PROGRAM GRAPH AND
1. . .   ABSTRACTLY REPRESENT AS A DMATRAN PROGRAM )
    . . INVOKE ( STRUC6 = USE STRUCTURED FORM REPRESENTATION TO REWRITE
1. . .   MODULE IN DMATRAN )
    . . INVOKE ( STRUC7 = PUNCH DMATRAN MODULE )
    . END IF
  END WHILE
END
```

Figure 4.1. Description of STRUCTRAN-2 Control

---

STRUCO consists of the main program for STRUCTRAN-2, all the routines it causes to be loaded, and all routines which are referred to directly or indirectly in two or more components. Since STRUC1 through STRUC7 communicate through the data base library, all data base service routines and storage manager routines are in STRUCO. Also all support routines performing commonly utilized functions are in STRUCO. The data base library is initialized by STRUC1. STRUC2 inputs the FORTRAN text to be structurized, adding it to the data base library after performing a lexical analysis to recognize FORTRAN key words and variables. STRUC3 reads the lexically analyzed text from the data base library and parses each statement, adding information on statement types and symbol usage to the data base library. STRUC4 utilizes all of the information accumulated so far in the data base library to identify the program graph of the module. This is added to the data base library in tabular form. STRUC5 performs a reduction-type analysis to recognize structured forms in the graph of the module. It updates the information on the data base library to indicate the structured form of the module. STRUC6 uses this representation to build the DMATRAN version of the original FORTRAN program. STRUC7 is used to print and punch statement text from the data base library. Since a single

program unit in UNIVAC 1108 FORTRAN may contain many routines (due to the internal subroutine capability), the iteration indicated in Fig. 4.1 is necessary to process modules consisting of more than one routine.

#### 4.3 STRUCTRAN-2 DATA STRUCTURES

STRUCTRAN-2 utilizes the data structures shown in Table 4.2. The FORTRAN source text to be structurized is input on LIN. A random-access data base library is created on LIBNEW. The FORTRAN source code is listed on LOUT, translated DMATRAN source code is written on LPUNCH, and a high-level audit trail is output to LDEBUG. The file activity performed by STRUCTRAN-2 was summarized in Fig. 2.1.

TABLE 4.2  
FILES USED IN STRUCTRAN-2 PROCESSING

Logical Unit Number	File Name	Data Structure	Mode	Storage Form	Record Format	Recommended Allocation
6LLIBNEW	LIBNEW	FORTRAN Program Description	Binary	Random	System Standard (installation-dependent)	Scratch file (mass storage)
6LLOUTPUT	LOUT	Reports	BCD	Sequential	Maximum 132 characters per line	System printer
6LLDEBUG	LDEBUG	Reports	BCD	Sequential	Maximum 132 characters per line	Scratch file (mass storage)
6LLPUNCH	LPUNCH	DMATRAN Translated Text	BCD	Sequential	Card image	System punch or permanent file*
5LINPUT	LIN	FORTRAN Text	BCD	Sequential	Card image	Card reader or permanent file*

\* Permanent files must have been previously created.

NOTE: The above files, along with other intermediate scratch files, are defined in subroutine FLINIT.

## APPENDIX A

### USING STRUCTRAN-2 ON THE UNIVAC 1108

Various modifications have been made to the UNIVAC 1108 installation of STRUCTRAN-2 to enhance its utility in the UNIVAC 1108 environment. The following points briefly summarize the enhancements:

- STRUCTRAN-2 accepts input either as source cards (run stream data) or as FORTRAN source elements referred to with the @ADD statement.
- STRUCTRAN-2 produces a program file in TPF\$ containing DMATRAN source code, translated FORTRAN source code, and FORTRAN relocatable elements for each program unit structurized.

As described in Sec. 2, no special control cards are required. Section 2 describes the use of STRUCTRAN-2, and illustrates STRUCTRAN-2 input and output.

The following run stream translates two FORTRAN source elements from program file PF2, assuming that the absolute element for STRUCTRAN-2 is on file PF1.

```
:      Assign permanent files
:
@XQT PF1.STRUCTRAN2
@ADD PF2.MAIN
@XQT PF1.STRUCTRAN2
@ADD PF2.XAMPLE
@COPY TPF$. ,PF3.
@FIN
```

STRUCTRAN-2 causes the STRUCTRAN-1 translator to process each DMATRAN element it outputs. A main program and subroutine XAMPLE are the FORTRAN source elements on program file PF2. The result of this run stream includes STRUCTRAN-2 FORTRAN listings of MAIN and XAMPLE, STRUCTRAN-1 indented listings of the structurized MAIN and XAMPLE, FORTRAN V listings of the translated FORTRAN versions for MAIN and XAMPLE, and a program file on PF3 containing the DMATRAN source code, the FORTRAN source code, and the relocatable elements for MAIN and XAMPLE. The DMATRAN source-code elements have MAIN and XAMPLE as element names; both have the version identifier STR. The first card image of DMATRAN source element MAIN is a DMATRAN control card (see STRUCTRAN-1 User's Manual, Appendix A) containing the name MAIN. This is followed by the DMATRAN source code for MAIN. The FORTRAN source code and relocatable elements have MAIN as the element name. Elements for XAMPLE are similar.



C	5100	DMPFRG	CHANGED MODULE NOT YET ON NEW LIBRARY.
C	5101	DMPFRG	ACTIVE CORE FRAGMENT NOT FOUND IN FDT.
C	5102	DMPMOD	MODULE NOT IN CORE TO DUMP.
C	5103	GETLHT	ILLEGAL TABLE NUMBER.
C	5104	GETLHT	ILLEGAL LIBRARY NUMBER.
C	5105	GETLHT	FRAGMENT BEYOND RANGE.
C	5107	GETMOD	MODULE NUMBER NOT YET ASSIGNED.
C	5108	GETMOD	MOD NOT ON LIBRARY.
C	5109	GETTAB	MODULE OUTSIDE LEGAL RANGE.
C	5110	GETTAB	MODULE NUMBER NOT YET ASSIGNED.
C	5112	LHOPEN	NO INFORMATION ON READ ONLY LIBRARY.
C	5113	LHREAD	ATTEMPT TO READ ILLEGAL LIBRARY IN LHREAD.
C	5114	LHWRT	ATTEMPT TO WRITE ON ILLEGAL LIBRARY IN LHWRT.
C	5115	MAKTAB	NO MORE TABLE SPACE AVAILABLE.
C	5116	MDBGET	MDBGET CAN ONLY BE USED WITH ACTIVE MODULES.
C	5117	MDBPUT	MDBPUT CAN ONLY BE USED WITH ACTIVE MODULES.
C	6200	DPSTMT	ILLEGAL DD-PATH NUMBER.
C	6201	ITPGHL	INDICATED STATEMENT HAS NO BLOCK STRUCTURE.
C	6400	WSGFRG	ILLEGAL CORE FRAGMENT NUMBER.
C	6401	WSGFRG	ILLEGAL LIBRARY FRAGMENT NUMBER.
C	6402	WSGWRD	ILLEGAL CORE ADDRESS.
C	6403	WSGWRD	ILLEGAL LIBRARY ADDRESS.
C	6404	WSPFRG	ILLEGAL CORE FRAGMENT NUMBER.
C	6405	WSPFRG	ILLEGAL LIBRARY FRAGMENT NUMBER.
C	7100	ACTDAT	DID NOT KNOW MODULE WAS ACTIVE.
C	7101	ACTFRG	NO SUCH FRAGMENT.
C	7102	ACTFRG	FRAGMENT NUMBER GIVEN AS ZERO.
C	7103	ACTMOD	ACTMOD DOES NOT KNOW AN ACTIVE MODULE IS
C			ACTIVE.
C	7501	ISIF	UNBALANCED PARENTHESES.
C	7502	GETFOF	TOO MANY CARDS.
C	7503	ISASGN	UNBALANCED PARENTHESES.
C	8100	NXTSTM	UNKNOWN LANGUAGE.
C	8101	BEGDDP	UNKNOWN LANGUAGE.
C	8102	FNDEND	NO END STATEMENT.
C	8103	POPDDP	POPDDP ERROR.
C	8104	PRSTMT	ILLEGAL CALL.
C	8105	STMOUT	TOO MANY DDPATHS BEGINNING AT ONE STATEMENT.
C	8106	SUBSTK	SUBSTK OVERFLOW.
C	8107	SUBPOP	SUBSTK UNDERFLOW.
C	8108	ADDJNC	OVERFLOW IN ADDJNC.
C	8109	INTRVL	OVERFLOW IN INTRVL.
C	8110	NAFT	LOOP WITH NO EXIT.
C	8111	NBEF	LOOP WITH NO EXIT.
C	8112	NOOSTK	NOOSTK OVERFLOW.
C	8113	INTSTK	INTSTK OVERFLOW.
C	8114	STRUCTX	UNKNOWN TYPE.
C	8115	TT	UNKNOWN TYPE.
C	8116	SESSTK	SSTACK OVERFLOW.
C	8117	PT	UNKNOWN TYPE.
C	8118	PRDSTK	SSTACK OVERFLOW.
C	8119	NEXIT	OVERFLOW IN NEXIT.
C	8120	NEXIT	UNDERFLOW IN NEXIT.
C	8121	JHVALU	RANGE ERROR IN INVALUE.
C	8122	ISRTAB	ILLEGAL TABLE NUMBER.
C	8123	ISRTAB	ISRTAB CANNOT BE USED WITH VARIABLE
C			LENGTH TABLES.
C	8125	ISRTAB	ILLEGAL MODULE NUMBER.
C	8126	STCK	STACK OVERFLOW
C	8127	DUPGEN	OVERFLOW IN DDPGEN
C	8128	DUPGEN	INFINITE LOOP FOUND

CCCCC~CC

C                    W A R N I N G   M E S S A G E S

C                    CCCCCCCCCCCCCCCCCCCCCCCCCCCCC

C    5101    STTBLK    DO TARGET IS NOT A CONTINUE STATEMENT.

C    7500    INDUP    ARWAY IN INDUP FULL.

C    7501    INDWN    ARWAY IN INDWN FULL.

CCCCC/CC

C                    F A T A L   E R R O R   M E S S A G E S

C                    CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

C    5000    FROM    ERROW LIMIT EXCEEDED.

C    5200    PUTWRU    SELECTED BLOCK OUT OF TABLE BOUNDS.

C    5201    PUTHLK    SELECTED BLOCK OUT OF TABLE BOUNDS.

C    5118    LHOPEN    OLD LIBRARY OPENED WITH IMPROPER FRAGMENT SIZE

C    5119    LHOPEN    NEW LIBRARY OPENED WITH IMPROPER FRAGMENT SIZE

C    5120    LHOPEN    R/W LIBRARY OPENED WITH IMPROPER PASSWORD

C    7500    GETFOR    \$ IN CONT. STMTS. MUST BE MANIALLY REMOVED.

## APPENDIX C

### HIGH-LEVEL AUDIT

STRUCTRAN-2 produces a high-level audit which may be printed by copying the normal scratch (unit 8) audit file to the printer. The audit describes the contents of the data base created and used by STRUCTRAN-2 in structuring a FORTRAN program, as well as an indented listing of the structured program. The high-level audit for the program in Fig. 2.2 is shown in Figs. C.1, C.2, C.3, and C.4. For an explanation of the reports generated, refer to the SDB, STB, and DDP Table definitions in STRUCTRAN-2 System Design and Implementation Manual (Sec. 3.3).

STATEMENT DESCRIPTOR BLOCKS FOR MODULE < SUM NUMBER >...

NO.	LABEL	KEYWORD	TYPE	WORD-PAIRS	(S)	IF PTR	LABEL PTR	PRIMARY NODE	AUX. NODE	STATIC COMPLEXITY
1		SUBROUTINE	200	9	1	0	0	0	0	0
2		INTEGER	1	5	14	0	0	0	0	0
3		ASSIGNMENT	100	3	29	2	0	0	0	0
4		IF-GOTO	200	8	35	7	0	0	0	0
5		ASSIGNMENT	100	3	51	2	0	0	0	0
6	50	ASSIGNMENT	100	5	57	2	9	0	0	0
7		IF	200	9	67	7	0	0	0	0
8		UN	100	7	85	11	0	0	0	0
9		ASSIGNMENT	100	4	99	2	0	0	0	0
10	100	ASSIGNMENT	100	5	115	2	14	0	0	0
11	10100	CONTINUE	200	1	125	4	11	0	0	0
12		IF-GOTO	200	4	127	7	0	0	0	0
13	999	RETURN	100	1	143	0	7	0	0	0
14		END	100	1	145	0	0	0	0	0

Figure C.1. Statement Descriptor Blocks for Module SUM

SYMBOL TABLE

NO.	NAME	CLASS	TYPE	MODE	BLUCK	DIM	1ST	2ND	3RD	1ST USE	TTL OCC	LAST USE	LAST INIT USE	EQ. PTD	EO. OFF	COM PTD	SR PTD	STM INTR	
< 1>	SUM	ENTRY	TYPELESS			3	14	0	3	0	0	0	0	UNUSED	0	0	0	0	14
< 2>	N	PARAMETER	VARIABLE	#UNKNOWN		0	14	0	3	0	0	0	0	UNUSED	0	0	0	0	24
< 3>	ADRAY	PARAMETER	ARRAY	INTEGER		0	14	0	3	0	1	2	0	UNUSED	0	0	0	0	24
< 4>	I<IM	PARAMETER	VARIABLE	#UNKNOWN		0	14	0	3	0	0	0	0	ASSIGNED	0	0	0	0	24
< 5>	I4	LOCAL	VARIABLE	INTEGER		0	14	0	3	2	1	2	0	UNUSED	0	0	0	0	24
< 6>	0	LOCAL	CONSTANT	#UNKNOWN		0	14	0	3	0	0	0	0	UNUSED	0	0	0	0	24
< 7>	999	LOCAL	VARIABLE	#UNKNOWN		13	14	0	3	4	2	13	0	USED	0	0	0	0	24
< 8>	J	LOCAL	VARIABLE	#UNKNOWN		0	14	0	3	0	0	0	0	ASSIGNED	0	0	0	0	24
< 9>	54	LOCAL	VARIABLE	#UNKNOWN		6	14	0	3	6	2	12	0	USED	0	0	0	0	24
< 10>	K	LOCAL	VARIABLE	#UNKNOWN		0	14	0	3	0	1	4	0	ASSIGNED	0	0	0	0	24
< 11>	1-100	LOCAL	VARIABLE	#UNKNOWN		11	14	0	3	0	2	11	0	DO-TARG	0	0	0	0	24
< 12>	1	LOCAL	VARIABLE	#UNKNOWN		0	14	0	3	0	1	4	0	ASSIGNED	0	0	0	0	24
< 13>	1	LOCAL	CONSTANT	#UNKNOWN		0	14	0	3	0	1	4	0	UNUSED	0	0	0	0	24
< 14>	1-0	LOCAL	VARIABLE	#UNKNOWN		10	14	0	3	10	1	10	0	UNUSED	0	0	0	0	24

Figure C.2. Symbol Table for Module SUM

```

-----
STMT LABEL STATEMENT TEXT***
-----
1 SUMMATIVE SUM ( N , ARRAY , ISUM )
2 INTEGER ARRAY ( 10 )
3 ISUM = 0
4 IF ( N .GT. 0 ) THEN
5   J = 0
6   DO UNTIL ( J .GE. N )
7     K = N - J
8     IF ( K .GT. 10 ) K = 10
9     I = 1
10    DO UNTIL ( I .GT. K )
11      ISUM = ISUM + ARRAY ( I )
12      J = J + 1
13      I = I + 1
14    END UNTIL
15  END IF
16  RETURN
17 END
-----

```

Figure C.3. DMATRAN Statements for Module SUM

DD-PATHS FOR MODULE SUM \*\*\*

NO.	1ST NOF	END NOF	EDG	CUM INO	PRU INO	TOP LVL	PAR DD	K ST	NO. ST.	DS INDEX	STATEMENTS ON DD-PATH
1	1	4	2	0	1	0	0	0	0	1	1
2	4	13	2	0	1	0	0	0	0	4	4
3	4	6	2	0	2	0	0	0	0	7	13
4	4	7	1	0	1	0	0	0	0	10	5
5	7	4	2	0	1	0	0	0	0	12	6
6	7	4	1	0	2	0	0	0	0	15	7
7	4	9	1	0	1	0	0	0	0	17	8
8	9	16	1	0	1	0	0	0	0	19	4
9	16	11	1	0	1	0	0	0	0	21	10
10	11	9	1	0	1	0	0	0	0	23	11
11	11	12	1	0	2	0	0	0	0	25	11
12	12	4	2	0	1	0	0	0	0	27	12
13	12	13	1	0	2	0	0	0	0	30	12
14	13	14	1	0	1	0	0	0	0	32	13
											14

Figure C.4. DD-Paths for Module SUM