

ZSTRMAC STRUCTURED MACRO DOCUMENTATION

12/30/78 DSH ORIGINAL DOCUMENTATION

05/16/06 DSH UPDATED FOR Z390 COMPATIBILITY WITH IBM TOOLKIT MACROS

TABLE OF CONTENTS

SECTIONS

1. INTRODUCTION TO STRUCTURED PROGRAMMING AIDS.
2. INTRODUCTION TO STRUCTURED MACRO ASSEMBLER.
3. GUIDE TO STRUCTURED MACROS.
4. SUMMARY OF STRUCTURED MACROS.
5. SYNTAX AND EXAMPLES OF EACH STRUCTURED MACRO.

INTRODUCTION TO STRUCTURED PROGRAMMING AIDS.

STRUCTURED PROGRAMMING IS THE GENERAL TERM APPLIED TO A NUMBER OF TECHNIQUES WHICH AID IN DEVELOPING PROGRAMS WHICH ARE EASIER TO READ, EASIER TO DEBUG, EASIER TO MAINTAIN AND WHICH ARE GENERALLY MUCH MORE RELIABLE.

STRUCTURED PROGRAMMING TOOLS DOCUMENTED HERE CONSIST OF THE EXTENDED CONTROL STRUCTURES NECESSARY TO WRITE TOP DOWN PROGRAMS WITH NO UNCONDITIONAL BRANCHES. THREE BASIC TYPES OF CONTROL STRUCTURES ARE IMPLEMENTED:

1. SEQUENTIAL CONCATENATION OF BLOCKS OF CODE.
2. SELECTION BETWEEN TWO OR MORE BLOCKS OF CODE.
3. REPETITION OF A BLOCK OF CODE.

FOR MORE INFORMATION ON THE THEORY OF STRUCTURED PROGRAMMING, SEE THE FOLLOWING REFERENCES.

DAHL, O. J., DIJKSTRA, E. W., HOARE, C. A. R., "STRUCTURED PROGRAMMING", ACADEMIC PRESS, LONDON (1972).

HIGGINS, D. S., "A STRUCTURED FORTRAN TRANSLATOR", ACM SIGPLAN, FEBRUARY (1975).

STEVENS, W. P., MYERS, G. J., CONSTANTINE, L. L., "STRUCTURED DESIGN", IBM SYSTEM JOURNAL 13, 2 (1974).

WEINBERG, G. M., "THE PSYCHOLOGY OF COMPUTER PROGRAMMING", VAN NOSTRAND REINHOLD, NEW YORK (1971).

YOURDON, EDWARD, "A BRIEF LOOK AT STRUCTURED PROGRAMMING AND TOP DOWN PROGRAM DESIGN", MODERN DATA, JUNE (1974).

INTRODUCTION TO ZSTRMAC STRUCTURED ASSEMBLER MACROS

THE ZSTRMAC STRUCTURED ASSEMBLER MACROS ALLOW ALC PROGRAMS TO BE WRITTEN IN A TOP DOWN STRUCTURED FORM WITHOUT THE NEED FOR ANY UNCONDITIONAL BRANCHES.

THE MACROS PROVIDE THE BASIC CONTROL STRUCTURES FOR ITERATION AND ALTERNATIVE SELECTION WITH A MINIMUM OF OVERHEAD AND FUNCTIONAL RESTRICTIONS.

THE DO WHILE=, DO UNTIL= AND ENDOD MACROS PROVIDE FOR ITERATION LOOPS.

## ZSTRDOC.TXT

THE IF, ELSEIF, ELSE, AND ENDF CONTROL MACROS PROVIDE FOR ALTERNATE SELECTION WITH COMPLEX TESTS. AND THE SELECT, WHEN, OTHERWISE, AND ENDSELECT MACROS PROVIDE FOR SELECTION OF MUTUALLY EXCLUSIVE ALTERNATIVES.

THE MACROS USE GLOBAL ARRAYS TO KEEP TRACK OF THE CURRENT LEVEL OF NESTING AND ANY REQUIRED GENERATED LABELS USED BY MULTIPLE MACROS WITHIN A CONTROL STRUCTURE.

GUIDE TO USING FPC STRUCTURED MACRO ASSEMBLER.

TO USE THE ZSTRMAC MACROS IN A Z390 ASSEMBLER PROGRAM, INCLUDE THE FOLLOWING COPY STATEMENT AT THE BEGINNING OF THE PROGRAM:

COPY ASMMSP

TO ASSEMBLE, LINK, AND EXECUTE THE DEMO\DEMOSTR1 EXAMPLE PROGRAM USING THE ZSTRMAC MACROS, USE THE FOLLOWING COMMAND FROM Z390 COMMAND LINE:

```
ASMLG DEMO\DEMOSTR1 SYSMAC(MAC+ZSTRMAC) SYSCPY(MAC+ZSTRMAC)
```

THE DEMOSTR1.MLC PROGRAM ILLUSTRATES USE OF DO WHILE, DO UNTIL, AND IF, ELSEIF, ELSE, ENDF STRUCTURES WITHOUT USING ANY REGISTERS. THE PROGRAM HAS PRINT NOGEN AT THE BEGINNING SO THE PRN LISTING IS NOT CLUTTERED WITH ALL THE GENERATED CODE. TO SEE THE GENERATED CODE LOOK AT THE GENERATED DEMOSTR1.BAL FILE OR COMMENT OUT THE PRINT NOGEN STATEMENT AND REASSEMBLE TO SEE THE GENERATED CODE IN THE PRN LISTING. THE EXECUTION LOG DEMOSTR1.LOG SHOWS THE WTO STATEMENTS EXECUTED IN THE SELECTED BLOCKS.

## SUMMARY OF STRUCTURED MACRO CONTROL STATEMENTS

### SELECTION OF ONE BLOCK FROM TWO OR MORE BLOCKS OF CODE

1. IF (TEST) - EXECUTE FOLLOWING BLOCK IF TEST IS TRUE OTHERWISE GO TO NEXT ELSEIF, ELSE, OR FI STATEMENT.
2. ELSEIF (TEST) - SAME AS ABOVE TO SELECT ONE OF A NUMBER OF BLOCKS.
3. ELSE - EXECUTE FOLLOWING BLOCK IF TEST IN PREVIOUS IF OR ELSEIF WAS FALSE.
4. ENDF - COMMON EXIT POINT FROM THE ABOVE CONTROL STRUCTURE.

### REPETITION OF A BLOCK OF CODE

1. DO WHILE=(TEST) - REPEAT BLOCK WHILE TEST IS TRUE
2. DO UNTIL=(TEST) - REPEAT BLOCK UNTIL TEST IS FALSE
3. ENDDO - EXIT POINT FOR ANY OF THE ABOVE DO CONTROL STRUCTURES.

## SYNTAX AND EXAMPLES

### DO UNTIL, (TEST)

FUNCTION - DEFINE BEGINNING OF BLOCK OF CODE WHICH IS EXECUTED REPETITIVELY UNTIL THE TEST IS TRUE. THE TEST IS PERFORMED AT THE END OF EACH EXECUTION OF THE BLOCK.

SYNTAX - SEE IF FOR TEST SYNTAX

EXAMPLE - DO UNTIL=(LTR, R1, Z, R1)  
 LR R2, R1  
 L R1, NEXT(R1)  
 ENDDO

DO WHILE=(TEST)

FUNCTION - DEFINE START OF BLOCK OF CODE WHICH WILL BE EXECUTED REPETITIVELY WHILE THE EXPRESSION IS TRUE. THE TEST IS PERFORMED BEFORE EACH EXECUTION OF THE BLOCK

SYNTAX - SEE IF FOR TEST SYNTAX

EXAMPLE - BRAS R12, GETREC  
 DO WHILE, (CLI, EOF, NE, TRUE)  
 BRAS R12, PROREC  
 BRAS R12, GETREC  
 OD

COMMENTS - THE ABOVE CONTROL STRUCTURE IS THE STANDARD FORM OF MOST FILE PROCESSING MAIN PROGRAMS. THE READ ROUTINE NAMED GETREC SETS EOF=FALSE WHEN END OF FILE OCCURS. THE ROUTINE NAMED PROREC PROCESSES RECORDS.

IF (TEST)

FUNCTION - DEFINE BEGINNING OF BLOCK OF CODE TO BE EXECUTED IF TEST IS TRUE. IF THE TEST IS FALSE, TRANSFER CONTROL TO THE NEXT ELSEIF, ELSE, OR ENDIF STATEMENT. IF CONTROL STRUCTURES MAY BE NESTED.

SYNTAX - TEST CONSISTS OF 4 OPERANDS SEPARATED BY COMMAS WHICH DEFINE INSTRUCTION, OPERAND 1, TEST CONDITION, AND OPERAND 2. MULTIPLE TEST MAY BE CONNECTED BY LOGICAL AND/OR KEYWORD CONNECTORS AS FOLLOWS:

IF (TEST), AND, (TEST), AND, (TEST), ...  
 IF (TEST), OR, (TEST), OR, (TEST), ...

IF 'AND' / 'OR' CONNECTORS ARE MIXED, IT IS INTERPRETED AS IF THERE IS AN EXTRA SET OF PARENTHESIS ADDED AFTER EACH CONNECTOR:

IF (TEST), AND, (TEST), OR, (TEST) MEANS

IF (TEST), AND, ((TEST), OR, (TEST)) ETC.

THE TEST CONDITION CODE VALUE MUST EQUATE TO 0-15 AND THE ZSTREQ.CPY MEMBER IS INCLUDED TO DEFINE EQUATES FOR ALL THE POSSIBLE CONDITION CODE VALUES.

FOR EXAMPLE:

IF (OP, OPR1, COND, OPR2)	OP OPR1, OPR2 BC 15-COND, TAGN
IF (SP, LINE, Z, =P' 0')	SP LINE, =P' 0'
BRAS R12, HEADING	BC 15-8, TAGN
FI	BRAS R12, HEADING

ZSTRDOC.TXT  
TAGN EQU \*

```
IF (CLC, KEY, L, =F' 10' )  
  BRAS R12, SMALL  
ELSEIF (CLC, KEY, L, =F' 50' )  
  BRAS R12, MEDIUM  
ELSE  
  BRAS R12, BIG  
ENDIF
```

ELSE

FUNCTION - DEFINE END OF BLOCK OF CODE EXECUTED IF PREVIOUS IF OR ELSEIF IS TRUE AND DEFINE BEGINNING OF BLOCK OF CODE TO BE EXECUTED IF ALL PREVIOUS IF AND ELSEIF STATEMENTS WERE FALSE.

SYNTAX - ALL KEYWORD

EXAMPLE - SEE IF

ELSEIF (TEST)

FUNCTION - DEFINE END OF BLOCK OF CODE EXECUTED IF PREVIOUS IF OR ELSEIF IS TRUE AND DEFINE BEGINNING OF BLOCK OF CODE TO BE EXECUTED IF EXPRESSION IS TRUE.

SYNTAX - SEE IF FOR TEST SYNTAX

EXAMPLE - SEE IF

ENDIF

FUNCTION - DEFINE END OF IF CONTROL STRUCTURE.

SYNTAX - NONE

EXAMPLE - SEE IF