THE UNIVERSITY OF MANITOBA

FORTRAN G COMPATIBLE CALCOMP PLOT ROUTINES FOR THE WATFIV COMPILER

by

Mark Steinert

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

WINNIPEG, MANITOBA

SEPTEMBER 1973

ABSTRACT

This thesis describes the modification of Calcomp Plot routines which run with WATFIV programs. These routines look identical to the Calcomp Plot routines for the FORTRAN G and H compilers. Plotting using Calcomp Plot routines becomes independent of the FORTRAN compiler used. A macro is also described which will generate WATFIV subprogram linkage code for an Assembler routine.

Acknowledgements

I would like to thank my thesis supervisor for his guidence and helpful suggestions during the preparation of this thesis.

Also, I would like to thank Dr. M. S. Doyle and Peter Pun for their time spent reading this thesis.

Table of Contents

Chapter 1

Introduction

This thesis is divided into six chapters. This chapter outlines the objectives of the thesis. CHAPTER 2 contains definitions and ancillary information on the WATFIV subprogram linkage conventions. Chapter 3 describes the WATSL macro, an aid to facilitate modification of any Assembler subprogram. Chapter 4 describes the modified Calcomp Plot routines. CHAPTER 5 describes how batch processing was achieved with the Calcomp Plot routines while chapter 6 sums up the work.

The object of this thesis is to provide a set of Calcomp Plot routines for WATFIV programs. These Calcomp Plot routines are described in (7). These routines are identical at the FORTRAN source level with the original Calcomp Plot routines which are used with the FORTRAN G and H compilers (4). Users may now use WATFIV to debug their plotting programs and then use FORTRAN G or H for production runs.

A FORTRAN IV plotting program, compiled by either of the FORTRAN G or H compiler, writes a set of data describing a plot on a plotter tape. Later the tape is mounted on the Calcomp 750/563 Incremental Drum plotter to draw the plot. Since this is an off-line system, each plot job requires computer operator assistance. This is the type of plotting

referred to in this thesis.

This thesis concerns the modification of the original Calcomp Plot routines so that they may be used by WATFIV. Much of this thesis concerns the problems encountered with the subprogram linkage and the programming techniques used to solve them. WATFIV subprogram linkage code (2) was added to the Calcomp routines so that WATFIV could check for proper subprogram usage.

This thesis assumes the reader has knowledge of FORTRAN IV, WATFIV, IBM S/360 Assembler, and OS subprogram linkage conventions.

Chapter 2

OS Subprogram Linkage Conventions

OS subprogram linkage conventions are used by the IBM System/360 Operating System and FORTRAN IV subprograms when compiled by either of the FORTRAN G or H compilers. These conventions are often used when programming in Assembler language (3) as well.

Subprogram referencing is accomplished by creating:

(i) A contiguous argument list consisting of addresses of the subprogram arguments

(ii) A save area to be used by the subprogram to store linkage information and register contents

(iii) A calling sequence for the calling program so that it may pass control to the subprogram.

It is the duty of the:

(i) calling program to provide a save area

(ii) called program to save any register contents and later restore them before returning.

The Contiguous Argument List

Each entry in the argument list is a word (four bytes)

of storage aligned on a fullword boundary. The first byte of the entry contains zeroes while the last three bytes contain the address of the argument. The last entry in the argument list has its sign bit set to a one. The address of the first word in this list is placed in register 1. This argument list is referred to in this thesis as the OS-type argument list.

The Save Area

The save area consists of 18 words (called SAVEAREA in this example) in the following format.

SAVEAREA	an internal address
(word 1)	which only WATFIV uses.
SAVEAREA+4	the save area address
(word 2)	of the calling program.
SAVEAREA+8	the save area address
(word 3)	of the called subprogram.
SAVEAREA+12	the contents of register 14.
(word 4)	
SAVEAREA+16	the contents of register 15.
(word 5)	
SAVEAREA+20	the contents of register 0.
(word 6)	

. .

. .

. .

SAVEAREA+68 the contents of register 12.

(word 18)

The address of the first word of the save area is placed in register 13 by the calling program.

The Calling Sequence

The addresses of the save area and the calling argument list are placed in the appropriate registers. The return address is saved in register 14. The address of the called program is placed in register 15 and a branch is made to that address.

The Assembler code shown below illustrates the OS subprogram linkage conventions used. This subprogram call passes two INTEGER*4 simple variables as arguments to the subprogram called RTN. The subprogram stores zeroes in these arguments and then passes these arguments back as it returns.

```
BALR 1,*+12          AN IN-LINE ARGUMENT
DC X'00',AL3(ONE)    IS CREATED AND
DC X'80',AL3(TWO)    POINTED TO BY REG 1
L 15,RTNAD           SUBPROGRAM ADDRESS
```

```
        BALR 14,15          CALL SUBPROGRAM
```

The called subprogram RTN.

```
        CSECT
        ENTRY RTN
RTN     STM 14,12,12(0,13)  SAVING REGISTER CONTENTS
        BALR 11,0
        USING *,11
        SR 2,2              CLEARING THE REGISTER
        ST 2,0(0,1)         STORING ZERO IN ONE
        ST 2,4(0,1)         STORING ZERO IN TWO
        BR 14               RETURNING
        DROP 11
        END
```

WATFIV Subprogram Linkage Conventions

The WATFIV subprogram linkage conventions are slightly different from the OS subprogram linkage conventions discussed previously. As a result, a WATFIV program will not run properly with an Assembler subprogram using OS linkage conventions. These linkage conventions are discussed in the following sections.

WATFIV Run-Time Routines

WATFIV provides a very extensive run-time error diagnostic facility. Run-time routines are called to check for any programming errors that can occur when control is transferred to a subprogram. More checking is performed before control is returned to the calling program. These checking operations are referred to as the entry sequence and return sequence operations. These operations are also referred to as the operations performed by WATFIV's prologue and epiloque (2). The run-time routines provide linkage and data for user programs and subprograms. Two additional routines are used when arrays are used as arguments to check for subscripting errors.

Most of these WATFIV run-time error checking routines are located in the control section labelled STARTA. Register 12 is used as the base register for addressing these routines. Since the storage space required by these routines is greater than the 4096 words addressed by register 12, an 'extended' run-time region labelled STARTB stores the remaining routines.

The Prologue and Epilogue

The prologue refers to the set of operations performed by the run-time routines to effect proper subprogram linkage when a subprogram is called. When a WATFIV calling program

and called subprogram are compiled, two argument lists are created. An actual argument list is created for the calling program and a dummy argument list is created for the called subprogram.

The following two WATFIV statements illustrate the FORTRAN statements that argument lists would be generated for.

In the calling program:

```
CALL SUB(A,B,6.)
```

In the called subprogram:

```
SUBROUTINE SUB(RA,RB,RC)
```

Whenever identifiers are used with no explicit type declaration accompanying them, the standard implicit type declaration is assumed.

The operations associated with the prologue are those which check the two argument lists for proper type compatibility, handle array size error checking, and copy the argument contents from the actual arguments of the calling program to the dummy arguments of the called subprogram.

The epilogue refers to the set of operations performed by other additional run-time routines which copy the contents of the dummy arguments back to the actual arguments

of the calling program and ensures that no constants, temporaries, DO-loop parameters, or assigned GO TO indices were modified by the called subprogram. Successful completion of these operations results in control being returned to the calling program.

WATFIV's call-by-value and call-by-location(name) Arguments

Call-by-value is the method most frequently used to pass arguments back and forth between programs. Using this method, the compiler run-time routines take the contents of an actual argument and copy it into the storage location reserved for the corresponding dummy argument. Simple variables and constants are usually handled in this way.

The other method employed by the WATFIV compiler is call-by-location. If slashes (ie. /) appear around dummy arguments in the subprogram declaration, they are treated as call-by-location arguments. The end result is that the compiler does not reserve any storage in the subprogram for these arguments. The addresses of the storage locations occupied by these arguments in the calling program are supplied to the subprogram. All references to these arguments are references to locations in the calling program. Array names used as dummy arguments are automatically treated as being call-by-location arguments. This results in no wasted duplication of storage space for

arrays. Subprogram names are also treated in this way.

When a return sequence operation is performed, the call-by-value dummy arguments are copied back into the storage locations of their corresponding arguments. The call-by-location arguments are not affected in any way by the return sequence operation. These concepts are discussed more fully in (1).

WATFIV Save Areas

A 24 word save area is used by WATFIV to store program linkage information and register contents. The format of the save area addressed by the label SAVEAREA in this example is illustrated below.

SAVEAREA	used by the internal
(word 1)	routines of the compiler.
SAVEAREA+4	the save area address
(word 2)	of the calling program.
SAVEAREA+8	the save area address
(word 3)	of the called subprogram.
SAVEAREA+12	the contents of registers
(words 4-18)	14,15,0,1,...,12.
SAVEAREA+72	The sign bit of this word is used as a
(word 19)	flag to indicate recursion. The other
	bits in this word are used by

	the compiler as a work area.
SAVEAREA+76	the contents of WATFIV's base
(words 20-24)	registers 5 to 10.

WATFIV uses 5 registers at run-time to refer to data areas internal to the compiler. These register contents are stored in the save areas so that they are always available to the run-time error checking routines.

WATFIV Argument Lists

A WATFIV argument list is set up in a contiguous vector aligned on a full word boundary. Each four byte word of the argument list is divided into a one byte field containing either a type code or a length specification (for hollerith strings) and a three byte address field. A type code is a numeric representation for the type of an identifier. A type code is represented here in hexadecimal form; written in Assembler as X'mn'. The address field contains either the address of an argument or a pointer to more information about that argument.

The different types of argument entries, the type codes associated with them, and their field formats are described below.

An argument entry is divided into the following six major categories:

(i) unchangeable quantity

(ii) variable, array element

(iii) array name

(iv) subprogram name

(v) statement number

(vi) argument list terminator.

Unchangeable Quantities

Any argument which should never be changed by the called subprogram is given this code. Examples of unchangeable quantities are constants, temporaries, DO-loop parameters, and assigned GO TO indices.

Their type codes have this basic format - X'0m' (in hexadecimal form) where m is a type number as listed in Table 1 on the following page.

The address field contains the address of the argument unless the identifier is of type CHARACTER*n where 1<=n<=7. In this case, the address field is AL3(QADDR) where QADDR is a full word of the form

```
QADDR      DC AL1(n),AL3(Q)
```

where n represents the length associated with this character variable and Q is the name of the character variable.

Table 1

Type numbers and s-values of WATFIV identifiers

Type numbers and s-values are numeric quantities used by the error checking routines in the compiler.

Identifier type	type number	s-value
Logical*4	0	2
Logical*1	1	0
Integer*4	2	2
Integer*2	3	1
Real*4	4	2
Real*8	5	3
Complex*8	6	3
Complex*16	7	4
Character*n n=1	8	0
Character*n n>1	9	0

Variables, Array elements

Their type code is X'8m' where m is a type number from Table 1. A simple variable, for example say V, is represented in the address field as AL3(V). An array element, say A, is represented as AL3(A). However if the argument is an array element, then an extra word follows the first one in the argument list. It has the following form.

```
DC X'8C',AL3(ASTAR)
```

The type code X'8C' indicates that this word is an additional word in the argument list. The identifier ASTAR represents the name of a star routine for the array A. Star routines are described in the section entitled 'Star Routines for Array Arguments'.

Example:

If A(3) is used as an argument, the corresponding argument list would be

```
          DC X'84',AL3(A1+8)   2 word offset from address
*                              of A.
          DC X'8C',AL3(ASTAR)  ASTAR represents the star
*                              routine for A
```

where A1 is used to represent the starting location of the array A.

Array Names

When an array name is used as an argument, the address field contains a pointer to the address of the star routine for this array and the type code is of the form X'km'. Here m represents the type number from Table 1 and k is 8+p where p is the number of dimensions the array possesses. For example, X'9m' is the type code for an array with one

dimension and X'Am' is the type code for an array with two dimensions. X'Fm' is the type code for an array with 7 dimensions.

Subprogram Names

When a subprogram name is passed to a called subprogram as an argument, its associated type code is X'50'. If the argument is a function, its associated type code is X'60'. In both cases, a pointer is stored in the address field pointing to a four byte word containing the subprogram's address.

Example:

The argument list entry for a SUBROUTINE subprogram called R is

```
DC X'50',AL3(RADDR)
```

where RADDR is illustrated below.

```
RADDR    DC A(R)
```

Statement Numbers

The type code associated with a statement number used as an argument is X'30'. For the statement number '&n' appearing in a calling argument list, the address field contains the address of the first machine language

instruction generated for the FORTRAN source statement numbered n.

Argument List Terminators

This argument entry marks the end of an argument list. It does provide some information about the expected nature of the called subprogram. If it is a SUBROUTINE subprogram, its type code is X'10' and if it is a FUNCTION subprogram, its type code is X'2m' where m is the type number found in Table 1 giving the type of the FUNCTION subprogram. The address field contains no information.

Example:

A FORTRAN call statement is illustrated below.

```
CALL PLUS(1,A(3),A,MINUS,&10)
```

The identifier A is dimensioned as a REAL*4 vector consisting of 10 elements. The identifier MINUS is the name of a SUBROUTINE subprogram which has been declared EXTERNAL in the calling program and 10 is a statement number used as an alternate return.

The WATFIV argument list for PLUS is translated into the following code by the WATFIV compiler.

```
DC X'02',AL3(ACONST)
DC X'84',AL3(A+8)
```

```
        DC X'8C',AL3(ASTAR)
        DC X'94',AL3(ASTAR)
        DC X'50',AL3(ATEMP)
        DC X'30',AL3(label for statement number 10)
        DC X'10',AL3(0)
```

In the calling program's data area, the following code is generated.

```
        DS 0F
ACONST  DC F'1'
A       DS 10F
ATEMP   DC A(MINUS)
        DC H'0',CL6'A'
ASTAR   BAL 15,XA1
        DC AL1(0),AL3(A)
        DC AL1(2),AL3(28)
        DC A(10)
```

ASTAR is the address of the star routine for A and is discussed thoroughly in the next section.

Star Routines for Array Arguments

The WATFIV compiler constructs a Subscript Testing and Addressing Routine (or Star routine) for each array in a WATFIV program. Each star routine contains an array's dimensions, starting address, and total length. At run-time

it is called upon to perform indexing operations to obtain an array element's contents, checking for out-of-range subscripts, and to provide the array's address to called subprograms.

A complete description of the star routines appears on pages 84 to 92 of the /360 WATFIV Implementation Guide (2). This section is taken from it with some modifications made.

The format of a full star routine known as ASTAR generated by the compiler for the array A is illustrated below.

```
            CNOP 0,4
            DC H'0',CL6'A'
ASTAR       BAL 15,xrtn    see note 1
            DC AL1(f),AL3(1 st element in array) see note 2
            DC AL1(s),AL3(length in bytes of array A)
            DC A(n)      see note 3
            DC A(d1)     see note 4
            DC A(d2)
             .
             .
            DC A(dk)
            DC B'C0C1C2C3C4C5C6C7',AL3(a1)    see note 5
            DC B'C00000000',AL3(a2)
            DC B'C00000000',AL3(a3)
             .
```

DC B'C00000000',AL3(aj)

where k - is # of dimensions declared for A.

j - is # of variable dimensions for A.

d1,d2, ... ,dk are the dimensions to be used for calculating the position of the element in the array.

a1,a2,...,aj are the variable dimensions used when declaring an array.

f = 4 * k - 4

s = s-value found in Table 1.

Note 1:

XA1 and XAN are the names of subscript test and evaluation routines internal to the WATFIV compiler in the STARTA control section. The symbol 'xrtn' stands for XA1 if k=1 or XAN if k>1.

Note 2:

If the array occurs in a subprogram's argument list, the prologue fills in the address for the first element from information supplied by the corresponding actual argument in the calling program's argument list. If the array has any dimensioning variables, the prologue fills in the dimension using the corresponding actual arguments.

Note 3:

This word is present only if the array is of type CHARACTER*n for n>1.

Note 4:

These address fields contain the dimensions to be used for calculating the position of an element in an array. For example, consider an array IBUF declared as

```
INTEGER*4 IBUF(10)
```

in a subprogram. The address field will contain the constant 10. Since there is only one dimension for IBUF, there will be only one four byte field containing the first dimension's size value. If the dimensioning variable used is not defined explicitly, the di, where i is the dimensioning variable's position, will contain zeroes until the prologue fills in this field. For example, consider an array IBUF declared as IBUF(NLOC). The address field will contain zeroes until the prologue fills this field in at run-time with the value of NLOC.

Note 5:

The symbols C0 to C7 represent bit positions. Bits C0 to C7 indicate which dimensions are variable. A bit value of 1 indicates a dimensioning variable. C1 is a 1 if the last dimension occurring in the subprogram declaration of

the array is variable. C2 is a 1 if the second last dimension is variable and etcetera. Bit C0 will be a 1 if the dimensioning variable occurring in the accompanying address field is in COMMON or is a call-by-location subprogram argument. Otherwise it is zero.

If none of the bits C1 to C7 are 1, then both C0 and a1 are zero. Otherwise C0 and a1 specify the location of the last dimension which is variable as follows.

If C0=0, then AL3(a1)=AL3(name of dimensioning variable).

If C0=1, then AL3(a1)=AL3(VBAR)

where VBAR is illustrated below.

```
VBAR     DC A(name of dimensioning variable)
```

The second, third,...,n words are present if there are 2,3,...,n dimensioning variables for n<=7. In this way, the second word indicates the second last dimension which is variable. The third word indicates the third last dimension which is variable. Bits C1 through to C7 of these words are set to zero. The C0 bit and ai field for 1<=i<=7 are interpreted according to the rules specified for the first set C0 and a1.

It is important to note that these words are not present if this is a star routine for an array which is not a subprogram's dummy argument.

The last operation performed by the prologue is to fill in the star routines for the dummy arrays. It uses the data described in note 4 to store information pertaining to the array and to calculate its total length.

This example taken from the /360 WATFIV Implementation Guide illustrates the Assembler code produced for a star routine for the array ALPHA.

The source statements are illustrated below.

```
      SUBROUTINE RTN(ALPHA,X,/M/,N)
      COMMON K
      DIMENSION ALPHA(10,K,N,5,M,12)
```

The star routine is illustrated below.

```
         CNOP 0,4
         DC H'0',CL6'ALPHA'
ALPHA    BAL 15,XAN
*  THE LOCATIONS CONTAINING *-* ARE FILLED IN BY THE
*  PROLOGUE.
         DC AL1(20),AL3(*-*) 1'ST ELEMENT'S ADDRESS
         DC AL1(2),AL3(*-*)  LENGTH
         DC A(10)
         DC A(*-*)  FILLED BY PROLOGUE FROM K
         DC A(*-*)  FILLED BY PROLOGUE FROM N
         DC A(5)
```

```
        DC A(*-*)   FILLED BY PROLOGUE FROM M
        DC A(12)
        DC B'10101100',AL3(MBAR)
        DC B'00000000',AL3(N)
        DC B'10000000',AL3(KBAR)
```

The variables MBAR and KBAR are kept in the subprogram's data area as is illustrated below.

```
KBAR    DC A(K)
MBAR    DC A(*-*)   FILLED BY PROLOGUE
```

The star routine ALPHA is also located in the subprogram's data area.

The complete set of argument types has been discussed here. Further discussion of argument types will only concern simple variables (INTEGER*4 or REAL*4), vectors (INTEGER*4 or REAL*4), and hollerith strings. These are the argument types which occur in the Calcomp plot routines.

A WATFIV Calling Sequence

The subprogram, say RTN, may be referenced in the following two ways depending upon whether it is a SUBROUTINE or a FUNCTION subprogram:

```
(i)  CALL RTN(ARG1, ARG2,...,ARGn)
(ii) ANY = RTN(ARG1, ARG2 ,...,ARGn)
```

where ARGi for 1<=i<=n are the arguments for the subprogram. A SUBROUTINE subprogram may have no arguments.

WATFIV generates the following calling sequences for either case.

```
        CNOP 2,4
        LA 14,ICI           RETURN ADDRESS
        L 15,=V(RTN)        CALLED PROGRAM RTN
        BALR 1,15
        DC AL1(C1),AL3(addr1)
        DC AL1(C2),AL3(addr2)
         .
         .
        DC AL1(Cn),AL3(addrn)
        DC AL1(Cn+1),AL3(0)
ICI     EQU *
```

For i=1,...,n,n+1, the Ci's are the type codes of the argument list entries and the addri's are the address of an argument or a pointer to more information. These fields were described in the section entitled 'A WATFIV Argument List'.

Immediately after the subprogram call has been made, the register contents are as follows:

- register 1 contains the address of the actual

argument list,

- register 13 contains the address of the 24 word save area,
- register 14 contains the return address, and
- register 15 contains the address of the subprogram's entry point.

The WATFIV compiler requires that floating point register 6 contain zeroes and register 12 contain the base address of the STARTA routine. Since the run-time routines must be available at all times, it is important to note that no Assembler subprogram should destroy the contents of register 12. It may store the contents away before it uses the register but these contents should be restored before returning control to a WATFIV compiled FORTRAN program.

ENTRY SEQUENCE CODE

The entry sequence code of a subprogram is the code produced to call the run-time error checking routine named XENT or the one named XENTSPEC. The following illustrates the entry sequence code produced for these WATFIV source statements.

```
      FUNCTION NAME(A,B)
      REAL*4 A,B

         CNOP 0,4
```

```
STM 14,11,12(13)
BAL 11,XENT (or XENTSPEC)
DC H'0',CL6'NAME'
DC A(SAVE)
DC AL1(84),AL3(A)
DC AL1(84),AL3(B)        model argument list
DC AL1(22),AL3(0)
....WATFIV will return control here following
    the execution of XENT or XENTSPEC.
```

XENT and XENTSPEC are routines in the STARTA control section for the run-time error checking routines. XENT contains a check to see if a subprogram is called recursively while XENTSPEC does not. These routines link the save areas in standard OS fashion. Register 11 points to the model argument list. The actual argument list pointed to by register 1 is compared against this model argument list by the prologue to ensure that the argument types match properly. The actual argument addresses or contents, depending upon the type of call, are then passed to the subprogram.

Return Sequence Code

The routine XRET located in STARTA performs the return sequence operations when returning control to a WATFIV calling program. The actions performed are basically the

inverse of the operations performed by the prologue. Arguments that were call-by-value types are passed back and checked for errors to ensure that the user hasn't modified an unchangeable quantity. The function result is loaded into register zero for a FUNCTION subprogram. Register 13 must point to the WATFIV save area that was used when the subprogram was called. It can then branch to the XRET routine. The XRET routine restores registers 1, 5 through to 11, 13, and 14. Register 12 always points to the STARTA routine.

The return sequence code is shown below.

```
BC 15,XRET   BRANCH TO XRET ROUTINE
```

If register 12 is modified and not reset properly WATFIV generates the following error message.

```
***ERROR*** COMPILER ERROR-EXECUTION TIME;RETURN TO SYSTEM
```

This is a general compiler error message used for many error situations and hence it's occurrence does not necessarily mean register 12 was modified.

Sample WATFIV program with an Assembler subprogram

A WATFIV program will run with an Assembler subprogram if an object deck produced from the Assembler subprogram is placed after the FORTRAN END statement and before the $ENTRY

control card.

Example:

The fcllowing sample program illustrates the use of concepts discussed previously. In this example, a WATFIV program calls an Assembler subprogram to print out a message.

```
//jobname JOB 'accounting information'
// EXEC WATFIV
//GO.OUTPUT DD SYSOUT=A
//GO.SYSIN DD *
$JOB  WATFIV
      I=3
      CALL TEST(I)
      PRINT,'I= ',I
      STOP
      END
        (object deck of TEST)
$ENTRY
$IBSYS
/*
```

The output listing is 2 pages with the following results displayed.

```
I= 0
```

```
THE CALL TO TEST WORKS FINE
```

The source listing of the Assembler subprogram TEST is:

```
TEST     CSECT
         USING R11,11
         CNOP 0,4
         STM 14,11,12(0,13)
         BAL 11,XENTSPEC(0,12)    BRANCH TO XENTSPEC RTN.
R11      DC H'0',CL6'TEST'
         DC A(SAVE)               WATFIV TYPE SAVE AREA
*  MODEL ARGUMENT LIST
         DC X'82',AL3(I)
         DC X'10',AL3(0)          LIST TERMINATOR WORD
*  XENTSPEC RETURNS CONTROL HERE
         OPEN (DCBOUT,OUTPUT) OPENING A DATA SET
         PUT DCBOUT,MSG           WRITING A MESSAGE
         CLOSE DCBOUT             CLOSING THE DATA SET
         BC 15,XRET(0,12)         BRANCH TO XRET ROUTINE
*  THE XRET ROUTINE WILL TRANSFER CONTROL BACK TO THE
*  CALLING PROGRAM.
         DS 0F
SAVE     DS 24F
I        DS F
DCBOUT   DCB DSORG=PS,RECFM=FB,MACRF=PM,LRECL=133,BLKSIZ
```

```
          E=133,DDNAME=OUTPUT
MSG       DC CL28' THE CALL TO TEST WORKS FINE'
          DC CL105' '
          END
```

In order to save using a BALR instruction, the

```
          USING R11,11
```

instruction is used. Register 11 still points to the entry sequence code after control returns from XENTSPEC and hence can be used as the base register for the rest of the program. Any register may be used as a base register if it is set up in the following way.

```
TEST      CSECT
          CNOP 0,4
            .                    as in the previous
            .                         example
          DC X'10',AL3(0)
          BALR 9,0
          USING *,9
```

The instruction

```
          BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC RTN.
```

causes control to branch to XENTSPEC. The XENTSPEC symbol used in the above instruction is equated to the value 246

and represents a displacement value. A displacement from base register 12 is used since XENTSPEC is not an entry point. If the STARTA routine is modified at any time, this displacement will also have to be modified to reflect this change. The instruction

```
XENTSPEC EQU 246
```

can be easily updated if it is required. XENTSPEC returns control to the instruction following the list terminator word for the model argument list.

The instruction

```
BC 15,XRET(0,12)
```

causes control to branch to the XRET routine. The XRET symbol is equated to the value 1124. Again a displacement is used because of the absence of an entry point for this routine.

Example:

The following sample WATFIV program is used to illustrate the linking conventions for array names.

```
DIMENSION IBUF(4000)
NLOC = 4000
CALL ARRAY(IBUF,NLOC)
PRINT,'NLOC IS ',NLOC
```

```
      STOP
      END
```

The following Assembler subprogram ARRAY prints out a message when it is called.

```
ARRAY    CSECT
         USING REG11,11
         STM 14,11,12(13)
         BAL 11,XENTSPEC(0,12)    BRANCH TO XENTSPEC RTN.
         DC H'0',CL6'ARRAY'
         DC A(SAVE)
*  THE MODEL ARGUMENT LIST
         DC X'92',AL3(IBUFST)
         DC X'82',AL3(NLOC)
         DC X'10',AL3(0)
         OPEN (DCBOUT,OUTPUT)
         PUT OUTPUT,MSG
         CLOSE DCBOUT
         BC 15,XRET(0,12)    BRANCH TO XRET ROUTINE
         DS 0F
SAVE     DS 24F
NLOC     DS F
DCBOUT   DCB DSORG=PS,RECFM=FB,MACRF=PM,LRECL=133,BLKSIZ
               E=133,DDNAME=OUTPUT
ITOKAY   DC CL29' THE CALL TO ARRAY WORKS FINE'
```

```
          DC CL104' '
          DS 0F
*  SETTING UP THE STAR ROUTINE
          DC H'0',CL6'IBUF'  NAME OF THE ARRAY
IBUFST    BAL 15,XA1(0,12)   BRANCH TO XA1
          DC AL1(0),AL3(*-*) FILLED IN
          DC AL1(2),AL3(*-*)     BY
          DC A(*-*)          PROLOGUE
*  REQUIRES THE FOLLOWING EXTRA WORD FOR THE VARIABLE
*  DIMENSION ELEMENT.
          DC B'01000000',AL3(NLOC)
          END
```

XA1 is a symbol equated to the value 1592. It is used as a displacement from register 12 to point to the XA1 routine in STARTA. XA1 is not an entry point in STARTA at present. This problem applies to XAN as well although it was never used in any of the subprogram modifications discussed in this thesis.

Chapter 3

The WATSL Macro

This macro generates Assembler code to provide WATFIV subprogram linkage for an Assembler subprogram. The types of arguments it generates WATFIV subprogram linkage for are simple variables, vectors, and hollerith strings. The following sections describe the macro's features, how it functions, and illustrates its use.

When called by an Assembler subprogram, the macro will generate the fcllowing in the order listed:

(i) entry sequence code to simulate a WATFIV subprogram argument list

(ii) star routines for any of the arrays occurring in the argument list

(iii) an OS-type argument list pointed to by register 1

(iv) storage locations for all the call-by-value arguments

(v) base register assignment instructions if requested.

The macro call is placed at the beginning of the Assembler subprogram. The code for these different phases or sections is produced in-line. Appropriate entry sequence

code is generated first for the subprogram's dummy arguments. Star routines are generated after the entry sequence code for any of the dummy arguments which are arrays. Even though code is added to allow WATFIV subprogram linkage, the subprogram requires the address of an OS-type argument list pointed to by register 1. This argument list is generated in-line using the information stored when generating the entry sequence code. Providing an OS-type argument list means the original subprogram may be left as it is with no extra modification other than providing the WATFIV entry and return sequence code.

When WATFIV executes the entry sequence code at run-time, it places the contents of the actual call-by-value arguments in storage locations reserved for the corresponding dummy arguments. Consequently, storage locations have to be generated for all the call-by-value arguments.

The macro user may use base register 11 or any base register that he specifies. Appropriate code is generated to initialize the base register following the entry sequence code.

Macro Parameters

The macro requires six parameters. These parameters are NAME, BASE, SAVE, ARGLIST, RTNTYPE, and ARRAY. They must

appear in the macro call in that order.

NAME - the name of the subprogram. This parameter must be a valid WATFIV identifier.

BASE - WATFIV's linkage convention is to use register 11 as its base register for the subprogram. Another base register may be specified here. Omitting this parameter indicates base register 11 will be used. It is the user's responsibility to drop register 11 as the base register when it is used. Specifying a register number or a register equated symbol results in base register assignment code being generated immediately following the code produced for the linkage.

SAVE - the name of a 24 word save area which will be used by WATFIV's prologue and epilogue when the program is executed. After executing the entry sequence code, register 13 contains the address of this save area.

ARGLIST - specifies the format of the subprogram's parameter list. It consists of a list of type and indicator codes (discussed shortly) separated by commas. The list is enclosed within parenthesis. The format of this list is described in more detail in

the section entitled 'The ARGLIST Field'.

RTNTYPE - a numeric value specifying the argument list terminator for the subprogram. For example, if the subprogram is an INTEGER*4 FUNCTION subprogram, the number 22 is used. If the subprogram is a SUBROUTINE, then the number 10 is used. The type code associated with FUNCTION subprograms was discussed in the 'Argument List Terminators' section.

ARRAY - omitted unless a hollerith string appears in the argument list for the subprogram. If this is the case, a positive integer is placed in the field as a flag indicating that additional code is required for a hollerith string. Only one hollerith string is permitted to appear in an argument list.

The ARGLIST Field

The ARGLIST field contains a sequence of indicator and type codes which will be called a parameter list in this section. An example of a parameter list consisting of only type codes would be (84,84,82). This parameter list generated by the macro would be identical to the one created for a WATFIV subprogram as is illustrated by this instruction.

SUBROUTINE PLOT(XPAGE,YPAGE,IPEN)

The indicator codes are numbers used to help describe certain types of subprogram arguments. They are used when vectors appear as arguments and require dimensioning variables. When such a vector appears as an argument, an INTEGER*4 dimensioning variable always appears in the argument list as well. WATFIV conventions require that this dimensioning variable be passed to the subprogram.

Before we proceed with further explanation, consider this example. The WATSL parameter list for the ARGLIST field describing a SUBROUTINE of the form

SUBROUTINE PLOTS(IBUF,NLOC)

where IBUF is a one dimensional array and NLOC represents the number of elements in that vector is (92,201,82)
where 92 - is the type code for an INTEGER*4 vector,
201 - is the indicator code for NLOC, the third element in the list, and
82 - is the type code for an INTEGER*4 variable.

Indicator codes are integers larger than 200. The mathematical result 'Indicator code - 200' gives the position of the vector name in the argument list. In the above example, the 1 indicates that the type code following it is the INTEGER*4 dimensioning variable for the first

argument. This mechanism achieves the same result as if the instruction

```
INTEGER IBUF(NLOC)
```

is available for use in the subprogram.

Any argument may be used to dimension any number of vectors. An indicator code for each vector is placed before the type code of the dimensioning argument.

An example of this useage is

```
SUBPROGRAM LINE(XARRAY,YARRAY,NPTS)
```

where XARRAY and YARRAY are vectors whose length is given by the argument NPTS. The parameter list (94,94,201,202,82) would appear in the ARGLIST field. The first indicator code 201 shows that NPTS is the dimensioning variable for XARRAY, while the indicator code 202 shows that NPTS is also the dimensioning variable for YARRAY.

An Example Using the WATSL Macro:

The following Assembler subprogram PLOTS illustrates the macro call used to provide WATFIV type linkage conventions that are similiar in effect to the code produced for the following two FORTRAN statements.

```
SUBROUTINE PLOTS(IBUF,NLOC)
```

```
      DIMENSION IBUF(NLOC)
```

This Assembler subprogram may be called by a WATFIV program once it is assembled into an object module. This subprogram stores the constant 2 in the first element of the vector passed as the first argument.

```
PLOTS    START
*  THE MACRO CALL FOLLOWS
         WATSL PLOTS,,SAVE,(92,201,82),10,
         L 2,0(0,1)          ADDRESS OF THE ARRAY
         LA 4,2              A 2 IS STORED IN
         ST 4,0(0,2)         THE FIRST ARRAY ELEMENT
         BC 15,XRET(0,12)    BRANCH TO XRET ROUTINE
         DS 0F
SAVE     DS 24F
         END
```

As this subprogram is being assembled, the WATSL macro call expands to become the following statements.

```
         USING R111,11
         CNOP 0,4
         STM 14,11,12(13)
         BAL 11,XENTSPEC(0,12)    BRANCH TO XENTSPEC RTN.
R111     DC H'0',CL6'PLOTS'
         DC A(SAVE)          WATFIV TYPE SAVE AREA
```

```
*  THE MODEL ARGUMENT LIST IS CREATED.
          DC X'92',AL3(STAR11)
          DC X'82',AL3(LOC13)
          DC X'10',AL3(0)
*  GENERATING CODE FOR THE STAR ROUTINES FOR THE ARRAY
*  NAMES.
          BC 15,*+32
          CNOP 2,4
          DC CL6'     '          ARRAY NAME IS LEFT BLANK
STAR      BAL 15,XA1(0,12)       BRANCH TO XA1
*  THE NEXT THREE ADDRESSES ARE CREATED AT RUN-TIME BY
*  THE WATFIV PROLOGUE.
LST11     DC AL1(0),AL3(0)       LST11 IS THE ADDRESS OF
          DC AL1(2),AL3(0)       THE ARRAY
          DC A(0)
          DC X'40',AL3(LOC13)
*  GENERATING THE OS-TYPE ARGUMENT LIST
*  TO BE USED BY THE SUBPROGRAM.
ABL1      BAL 1,*+4*3            AN IN-LINE ARGUMENT LIST
          DC X'00',AL3(0)        FILLED IN LATER BY A
          DC X'80',AL3(LOC13)    MVC INSTRUCTION
*  A MVC  INSTRUCTION MOVES AN ARRAY ADDRESS INTO THE
*  OS-TYPE ARGUMENT LIST AT RUN-TIME.
          MVC ABL1+4*1+1(3),LST11+1
          CNOP 0,4
```

```
*  GENERATING THE STORAGE LOCATIONS FOR THE
*  CALL-BY-VALUE ARGUMENTS.
         BC 15,*+4*(3-1-0)
LOC13    DS F
```

The prologue fills in all the storage locations for the call-by-value dummy arguments with the contents of the actual arguments. However the call-by-location array names can only be obtained from their respective star routines and only after the prologue has moved the address to the subprogram. The array address is obtained from the star routine in the subprogram and inserted into the OS-type argument list by a MVC instruction after the prologue completes its operations. In the sample subprogram, the prologue places the address of the vector IBUF in the location designated as LST11. A MVC instruction following the code generated for an OS-type argument list moves a copy of the vector's address into the appropriate position in the OS-type argument list.

All addresses of call-by-value arguments found in the OS-type argument list refer to storage locations generated for them by the macro call. Some duplication exists in the subprogram. However, the organization is kept clear by this method.

A listing of the WATSL macro appears in Appendix A.

Chapter 4

Implementing a WATFIV Version of the Calcomp Plot Routines

The Calcomp plotting routines referred to in this thesis consist of the subroutines which make up the Basic Software package provided by California Computer Products, Inc. and are described in (8). These routines are PLOTS, SYMBOL, NUMBER, SCALE, AXIS, and LINE. The subroutine called PLOTS has four additional entry points for PLOT, OFFSET, FACTOR, and WHERE. The University of Manitoba provides 2 additional subroutines with this package. These routines are SIMBLE and AXISI and are modifications to SYMBOL and AXIS respectively. The University of Manitoba publishes a programmers guide (7) which is a modification of (8) and contains additional information on SIMBLE and AXISI. The routines PLOTS, SYMBOL, SIMBLE, and NUMBER are written in Assembler language while the rest are written in FORTRAN IV. This chapter describes the modifications to these routines in order for them to run with WATFIV.

The subprogram calling relationship for these routines follows.

SUBPROGRAM	USES
PLOTS	none
SYMBOL	PLOT
NUMBER	SYMBOL
SCALE	NONE
LINE	PLOT
	SYMBOL
AXIS	PLOT
	SYMBOL
	NUMBER

PLOTS

This section describes the Assembler code for the PLOTS routine before and after its modification to run with the WATFIV compiler. A discussion of the coding modifications follows.

The start of the original PLOTS routine for the FORTRAN G and H compilers is illustrated below.

```
PLOTS    START 0
         ENTRY PLOT
         ENTRY FACTOR
         ENTRY WHERE
         ENTRY OFFSET
         BCR 0,0
```

```
        STM 14,12,12(13)
        LR 11,15
        LA 12,SAVE
        ST 12,8(0,13)
        ST 13,SAVE+4
        LR 13,12
        LE 0,STPSZ
        ME 0,FACT
        STE 0,FACSZ
        MVI PLOT(4),PLOTS+2
        SR 0,0
        .
        .
```

The code produced for the PLOTS routine in order to modify it for WATFIV subprogram linkage is illustrated below.

```
PLOTS   START 0
        ENTRY PLOT   WATFIV TYPE ENTRY POINT
        ENTRY PLOT1  OS-TYPE ENTRY POINT
        EXTRN ERROR1 WATFIV ERROR MESSAGE ROUTINE
        ENTRY FACTOR
        ENTRY WHERE
        ENTRY OFFSET
        WATSL PLOTS,,SAVE,(92,201,82),10,
```

```
AXFZ     BCR 0,0  TO BE MODIFIED TO PREVENT REENTRY
         BCR 0,0
         CNOP 0,4
*  THIS INSTRUCTION MOVES CODE TO ALLOW
*  AN OS-TYPE ENTRY TO PLOT.
         MVC PLOT1(2),AXFZ
         MVC ALIZ(4),AXFZ ALLOWS WATFIV ENTRY TO PLOT
         MVC AXFZ(4),AGBZ PREVENTS REENTRY TO PLOTS
         LE 0,STPSZ
         ME 0,FACT
         .
         .
```

The WATSL macro copies the actual arguments into the called subprogram.

Since the BASE parameter was omitted in the macro call, register 11 is used as the base register. The code which performed the OS-type register save operations was removed and replaced by the macro call. It then becomes WATFIV's responsibility to link the save areas and save the contents of the registers.

The PLOTS routine must be called prior to calling any of the other Calcomp plotting routines which call PLOT. When PLOTS is executed, the instructions

```
PLOT     BCR 15,14  PREVENTS INVALID ENTRY
```

```
        BCR 15,14  IS MODIFIED AT RUN-TIME
```

which prevent entry to PLOT are changed to two

```
        BCR 0,0
```

instructions and the original instruction

```
        BC 15,XRET(0,12)
```

is placed in the PLOTS routine thereby preventing more than one access to PLOTS. The code following the OS-type entry point is also modified to allow entry to the PLOT routine.

PLOT

The original code for PLOT is illustrated below.

```
        USING *,11
PLOT    BCR 15,14  PREVENTS INVALID ENTRY
        BCR 15,14  IS MODIFIED AT RUN-TIME
*  A BASE REGISTER IS SET UP AND SAVE AREA PROVIDED
        LR 11,15
        LA 12,SAVE
        ST 12,8(0,13)
        ST 13,SAVE+4
        LR 13,12
        LM 2,4,0(1)
        .
```

The code after modification is illustrated below.

```
PLOT     WATSL PLOT,,10,SAVE,(84,84,82),10,
*  MODIFIED BY PLOTS TO ALLOW ENTRY TO PLOT
ALIZ     BC 15,XRET(0,12)     BRANCH TO XRET ROUTINE
         BC 15,PLOT2          IT IS MODIFIED AT RUN-TIME
*  CODE FOR AN OS-TYPE ENTRY POINT.
PLOT1    BR 14                MODIFIED BY PLOTS
         STM 14,12,12(13)
         LA 3,8               ADJUSTING THE BASE REG
         SR 15,3              REG 15 MUST ALWAYS POINT
         LR 10,15             TO PLOT1
         LA 9,SAVE            OS-TYPE SAVE CONVENTIONS
         ST 9,8(13)
         ST 13,SAVE+4
         LR 13,9
         LA 2,1               SETTING THE ENTRY
         ST 2,ENTFLG          SEQUENCE FLAG NEGATIVE
P LOT2    LM 2,4,0(1)         LOAD LINKAGE
          .
          .
          .
```

PLOT1 is an entry point using OS-type linkage conventions for the PLOT routine. This entry point enables the SYMBOL and SIMBLE routines to call PLOT using OS subprogram linkage conventions by referring to PLOT1. A

WATFIV calling sequence is not required for calling PLOT.

The return sequence code is illustrated as follows.

```
EXIT     STM 1,3,BUFID       SAVE BUF INFORMATION
*  DECIDING WHICH RETURN SEQUENCE TO USE
*  A WATFIV RETURN, OR AN OS-TYPE RETURN.
         L 3,ENTFLG          ENTRY SEQUENCE FLAG
         LTR 3,3                   ZERO ?
         BC 8,WATRET         WATFIV RETURN SEQUENCE
         SR 3,3              NO.
         ST 3,ENTFLG         RESETTING FLAG
         L 13,SAVE+4
         LM 14,12,12(13)
         BR 14               OS-TYPE RETURN
*  A WATFIV TYPE RETURN
WATRET   BC 15,XRET(0,12)    BRANCH TO XRET ROUTINE
```

This code checks an entry sequence flag to see whether a WATFIV return sequence or an OS-type return sequence should be performed.

FACTOR

The following illustrates the code for the FACTOR routine before any modifications were made.

```
          USING *,15
FACTOR    ST 2,SAVE
          L 2,0(1)              LOAD LINKAGE
          .
          .
          .
          STE 0,FACSZ
          L 2,SAVE
          BCR 15,14
          DROP 15
```

The following illustrates the code after modifications were made.

```
FACTOR    WATSL FACTOR,9,SAVE,(84),10
          L 2,0(1)              LOAD LINKAGE
          .
          .
          STE 0,FACSZ
*  A WATFIV TYPE RETURN
          BC 15,XRET(0,12)      BRANCH TO XRET ROUTINE
          DROP 9
```

Once this routine has calculated the new factor size, it executes the WATFIV return sequence code to return control to the calling program. The base register that was used was dropped at the end of the routine.

WHERE AND OFFSET

The modifications to WHERE and OFFSET are very similar to those performed on the FACTOR routine. Listings of these modified routines appear in Appendix B.

SYMBOL

The standard calling sequence for SYMBOL is

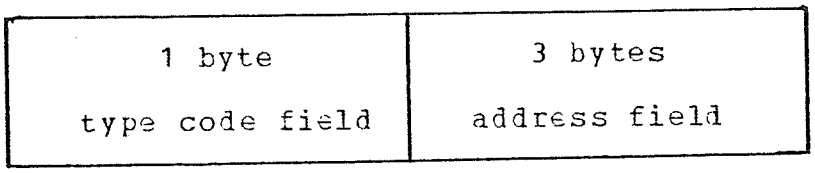
```
CALL SYMBOL(XPAGE,YPAGE,HEIGHT,IBCD,ANGLE,NCHAR)
```

where the arguments are interpreted according to descriptions in (7).

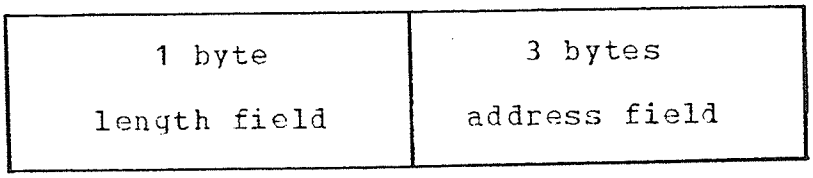
The argument IBCD contains text in BCD or A-type format. These characters may be stored in a simple variable (INTEGER*4 or REAL*4), a vector (INTEGER*4 or REAL*4), or a hollerith string. The FORTRAN G and H compilers pass the address of the first location of this argument as the IBCD argument. The number of characters in this argument is given by the argument NCHAR. Error checking, to ensure that the user does not use more characters than have been

provided, does not exist.

WATFIV uses a different linkage convention in its argument list to accomodate vector and hollerith string arguments. A vector results in one word being used in the calling list. Its type code is either X'92' (INTEGER*4 vector) or X'94' (REAL*4 vector). The following 3 byte argument list entry contains a pointer to a star routine for this element. The hollerith string (also referred to as a character constant) is represented by 2 words although only one word appears in the argument list. The second word is a 'dope vector' and is stored in the program's data area. A schematic illustration of the storage format for a character constant appears below.



This word appears as an entry in an argument list. The type code for a character constant is X'09'. The address field contains a pointer to a dope vector.



This word appears in the program's data area. The length field specifies the length of the character constant. The address field contains the address of the character constant. The character constants are stored in a contiguous list of words in the program's data area. Blanks are inserted on the right if the character constants do not occupy multiples of full words.

When a WATFIV mainline program makes the following call to SYMBOL

```
CALL SYMBOL(XPAGE,YPAGE,HEIGHT,IBCD,ANGLE,NCHAR)
```

the following calling sequence is generated.

```
        CNOP 2,4
        L 3,=V(SYMBOL)       SUBROUTINE ADDRESS
        LA 14,ICI            RETURN ADDRESS
        BALR 1,3
        DC X'84',AL3(XPAGE)
        DC X'84',AL3(YPAGE)
        DC X'84',AL3(HEIGHT)
        DC X'80',AL3(0)      THIS TYPE CODE AND
        DC X'84',AL3(ANGLE)  ADDRESS FIELD DEPEND
        DC X'82',AL3(NCHAR)  ON TYPE OF ARGUMENT
        DC X'10',AL3(0)      USED FOR IBCD.
ICI.....
```

If IBCD is a simple INTEGER*4 variable, the type code and address field are:

```
DC X'82',AL3(IBCD)
```

If IBCD is an INTEGER*4 vector, the type code and address field are

```
DC X'92',AL3(STAR11)
```

where STAR11 is the name of the star routine for IBCD. If IBCD is of type REAL*4, the type code is X'84' for a simple variable and X'94' for a vector. If IBCD is a character constant such as 'QUOTE ', the argument list entry is

```
DC X'09',AL3(DOP11)
```

where X'09' is the type code for a character constant and DOP11 occurs in the program's data area as follows.

```
DOP11   DC X'06',AL3(label for the character constant)
```

The address field contains the address of the character constant. The character constant occupies 2 words of storage and is in the following form.

```
DC CL8'QUOTE   '
```

Examples in Chapter 3 showed the code generated by the WATSL macro when simple variables and vector names are used

as arguments. The following example illustrates the code produced when a character constant is among the list of arguments.

The macro call

```
          WATSL SYMBOL,,SAVE,(84,84,84,9,84,82),10,
```

generates the following code.

```
          USING R111,11
          CNOP 0,4
          STM 14,11,12(13)
          BAL 11,XENTSPEC(0,12)    BRANCH TO XENTSPEC RTN.
R111      DC H'0',CL6'SYMBOL'
          DC A(SAVE)
*  THE MODEL ARGUMENT LIST IS CREATED.
          DC X'84',AL3(LOC11)
          DC X'84',AL3(LOC12)
          DC X'84',AL3(LOC13)
          DC X'09',AL3(DOP11) ADDRESS OF DOPE VECTOR
          DC X'84',AL3(LOC15)
          DC X'82',AL3(LOC16)
          DC X'10',AL3(0)
*  GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY
*  THE SUBPROGRAM.
ABL5      BAL 1,*+4*7
          DC X'00',AL3(LOC11)
```

```
          DC X'00',AL3(LOC12)
          DC X'00',AL3(LOC13)
DOP51     DC X'00',AL3(0)     DOPE VECTOR IS FILLED
          DC X'00',AL3(LOC15) IN AT RUN-TIME
          DC X'80',AL3(LOC16)
*  GENERATING THE STORAGE LOCATIONS FOR THE
*  CALL-BY-VALUE ARGUMENTS.
          BC 15,*+4*(7-0-1)
LOC11     DS F
LOC12     DS F
LOC13     DS F
LOC15     DS F
LOC16     DS F
```

The macro generates the code for the character constants treating them as if they were call-by-location arguments. The dope vector for the character constant shown in the example above, is located in the OS-type argument list created for the assembler subprogram. The dope vector is filled with zeroes at compile time since the length and address of the character constant are unknown to the subprogram at that time. However at run-time when control is passed to the subprogram, the dope vector of the calling argument list is available. The following two instructions

```
          L 2,12(0,1)
```

```
MVC DOP11(1),0(2)
```

place the length of the character constant in the length field of the subprogram's dope vector. These two instructions are executed prior to executing the subprogram's entry sequence code. When the entry sequence code is executed, the run-time error checking routine XENTSPEC checks that the two character constants are the same length. They are found to be exactly the same of course, and then the address of the character constant is placed in the address field of the subprogram's dope vector. This is the technique used to pass character constants to a subprogram.

One should note that in order to save a storage location, the dope vector is placed in the OS-type argument list for the called subprogram. This may be a problem for subprograms which check for the end of the argument list. If a character constant greater than 127 characters is passed to a subprogram, the dope vector's sign bit will be interpreted as being set on. This is because the length field is occupying the byte of storage which normally contains X'00'. For example, a character constant consisting of 129 characters would have a X'81' stored in the length field of its dope vector. This word would be interpreted as being the end of the OS-type argument list. If this is the case, this byte should be set to zero by a

```
MVI DOP51,X'00'
```

instruction after the entry sequence code. Since SYMBOL does not check for the end of the argument list, this byte is not modified.

If the IBCD argument in the SYMBOL call is a vector of type INTEGER*4 or REAL*4, an additional problem is encountered. Since this vector can be of any length, WATFIV expects a dimensioning variable to be passed to the subprogram in addition to the vector. This is not the case when a call to SYMBOL is made with the original version that was compiled by the FORTRAN G and H compilers. In that case, the address of the vector is the only information passed about the vector. WATFIV requires the dimensioning variable in order to construct a star routine in the subprogram for this vector. This dimensioning variable is not available in a call of the form

```
CALL SYMBOL(XPAGE,YPAGE,HEIGHT,IBCD,ANGLE,NCHAR)
```

which is the calling form of the original version. One solution tried was to attach this dimensioning variable to the end of the argument list as an extra argument. This approach works in practice. A plotting program containing a call to SYMBOL with the extra argument attached to the argument list would run with WATFIV and would also run with the FORTRAN G and H compilers since the SYMBOL routine

ignores the extra argument at the end of the argument list. The disadvantage with this approach is that a program written for the FORTRAN G or H compiler that does not use the extra argument for the SYMBOL calls, will not work when run with WATFIV. Therefore the approach used was to leave the argument list as it was and to obtain the vector length at run-time from the actual argument list. This list contains an entry for the IBCD vector whose address field contains a pointer to the vector's star routine. This star routine contains a 4 byte word containing the vector's length. When control enters the SYMBOL routine at run-time, the following instructions are executed to obtain this vector length and copy it into a storage location whose address is placed in the star routine for the dummy vector argument found in the called subprogram's argument list. These instructions are:

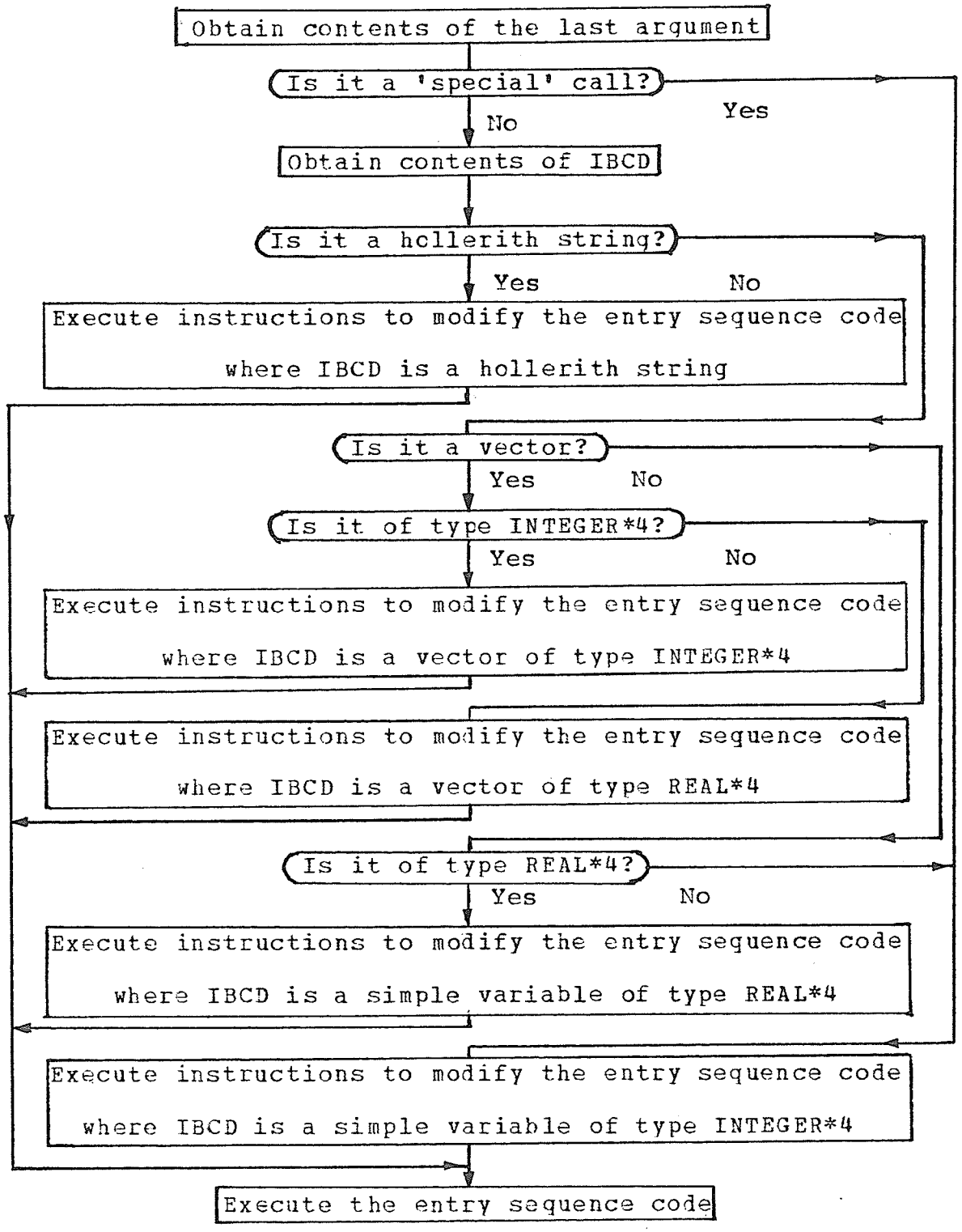
```
L 2,12(0,1)          IBCD'S STAR ROUTINE
L 2,12(0,2)          OBTAIN THE VECTOR'S LENGTH
ST 2,TPST1
```

The star routine for the dummy vector for IBCD contains a word containing the address of the dimensioning variable. The macro generates this storage location. It is called TPST1 in this example. These instructions are executed before the entry sequence code is executed. This approach

allows a vector to be used as an argument with the full power of variable dimensioning but without the additional argument to serve as its dimensioning variable. This technique is used again for the SCALE, LINE, and AXIS routines.

The two types of calls to SYMBOL are the 'standard' call and the 'special' call. The 'standard' call is used to draw text such as titles while the 'special' call is used to draw special centered symbols for plotting data points. A call is a 'special' call if and only if the last argument has a negative value. The fourth argument (ie. INTEQ on page 11 of (7)) in the argument list for SYMBOL is always treated as an INTEGER*4 variable for 'special' calls. For a 'standard' call, the argument (ie. now called IBCD) may be one of a simple variable (INTEGER*4 or REAL*4), a vector (INTEGER*4 or REAL*4), or a hollerith string. Consequently, the IBCD argument may be one of five different types. Since the subprogram has to accommodate each different type of argument it is expecting, the approach taken was to modify the entry sequence code at run-time according to the type of the SYMBOL call and the IBCD argument. The type of the SYMBOL call is determined by examining the last argument in the actual argument list. The type of the IBCD argument is determined by examining the type code of the corresponding actual argument.

The fcllowing flow chart illustrates the program logic required to execute the instructions which modify the entry sequence code appropriately.


Obtain contents of the last argument
Is it a 'special' call?
Yes
No
Obtain contents of IBCD
Is it a hollerith string?
Yes
No
Execute instructions to modify the entry sequence code where IBCD is a hollerith string
Is it a vector?
Yes
No
Is it of type INTEGER*4?
Yes
No
Execute instructions to modify the entry sequence code where IBCD is a vector of type INTEGER*4
Execute instructions to modify the entry sequence code where IBCD is a vector of type REAL*4
Is it of type REAL*4?
Yes
No
Execute instructions to modify the entry sequence code where IBCD is a simple variable of type REAL*4
Execute instructions to modify the entry sequence code where IBCD is a simple variable of type INTEGER*4
Execute the entry sequence code

Use of SIN/COS Functions

SYMBOL calls the SIN and COS functions. These functions occur as entry points in the IHCSSCN control section. The original SYMBOL routine calls the SIN and COS subprograms that are stored in the FORTRAN IV library. These library subprograms use OS subprogram linkage conventions. In order to call the SIN and COS functions in the WATFIV library, a WATFIV calling sequence has to be set up. This calling sequence for a library subprogram is different from that used to call WATFIV subprograms. Therefore the SIN and COS routines from the FORTRAN IV library were modified so that they could be called by SYMBOL using OS-subprogram linkage conventions. This involved removing an error message request which referenced the two entry points IHCERRM and IBCOM# which were not available in the WATFIV library. These entry points handle error message requests for subprograms written using OS subprogram linkage conventions. This error message request was removed and a call was made to the WATFIV subprogram called ERROR1 to print out an error message. The error message follows.

```
***ERROR*** ABS(SIN OR COS FUNCTION ARGUMENT) >= PI *2**18.
```

The entry points for the modified IHCSSCN control section were changed to SIN1 and COS1 so that no confusion would occur with WATFIV SIN and COS library subprograms.

The SYMBOL routine was modified to call SIN1 and COS1.

The code inserted into the SYMBOL routine to handle the different types of calls and the different types of arguments allowed is illustrated below.

```
SYMBOL   START 0
         EXTRN SIN1          SIN1 AND COS1 ARE THE
         EXTRN COS1          SIN AND COS FUNCTIONS
         EXTRN PLOT1         OS-TYPE ENTRY POINT
         ENTRY SYMB1         OS-TYPE ENTRY POINT
         BALR 9,0
         USING *,9
*  CHECKING TYPE OF CALL MADE TO SYMBOL
         L 2,20(0,1)         ADDRESS OF ICODE
         L 2,0(2)            OBTAIN CONTENTS OF ICODE
         LTR 2,2             IS IT A 'SPECIAL' CALL
         BC 4,INTS           BRANCH IF SO
*  IT WAS A 'STANDARD' CALL
*
*  DETERMINING WHETHER IBCD IS A SIMPLE VARIABLE,
*  A VECTOR, OR A HOLLERITH STRING.
         TM 12(1),X'09'      HOLLERITH STRING ?
         BC 1,CHARK          YES
         TM 12(1),X'90'      VECTOR?
         BC 12,NEXTY         NO. IT IS SIMPLE VARIABLE
*  IBCD IS A VECTOR
```

```
         L  2,12(0,1)          IBCD'S STAR ROUTINE
         L  3,4(0,2)           THE CALLING VECTOR
         L  2,12(0,2)          VECTOR LENGTH OF ARGUMENT
*  IT IS STORED IN A LOCATION REFERENCED BY THE STAR
*  ROUTINE FOR THE DUMMY ARGUMENT REPRESENTING IBCD.
         ST 2,TPST
*  STORING THE VECTOR ADDRESS IN THE OS-TYPE
*  ARGUMENT LIST.
         ST 3,DOPE
         LA 2,STAR             ADDRESS OF STAR ROUTINE
         ST 2,IBCD             STORING THIS ADDRESS
*
*  DETERMINING TYPE OF IBCD ARGUMENT
         TM 12(1),X'02'
         BC 1,INTV
         MVI IBCD,X'94'        IS REAL*4 TYPE
         BC 15,ENTSEQ
INTV     MVI IBCD,X'92'        IS INTEGER*4 TYPE
         BC 15,ENTSEQ
*  IBCD IS A HOLLERITH STRING
CHARK    L 2,12(0,1)           ADDRESS OF DOPE VECTOR
*  MOVING THE VECTOR LENGTH INTO THE DOPE VECTOR
*  LENGTH FIELD OF THE DUMMY ARGUMENT FOR IBCD.
         MVC DOPE(1),0(2)
         LA 2,DOPE
```

```
         ST 2,IBCD           ADDRESS OF DOPE VECTOR
         MVI IBCD,X'09'      SETTING THE TYPE CODE FOR
         BC 15,ENTSEQ        THE IBCD ARGUMENT
*  HAVE A 'SPECIAL' CALL
INTS     LA 2,1              SETTING THE
         ST 2,SPECFLG        SPECIAL FLAG ON
*IBCD IS A SIMPLE VARIABLE
NEXTY    LA 2,TPST           IBCD IS TREATED AS BEING
         ST 2,IBCD           A CALL-BY-VALUE ARGUMENT
         ST 2,DOPE
*
*  CHECKING FOR A 'SPECIAL' CALL
         L 2,SPECFLG
         LTR 2,2
         BC 2,ISIMP1         IT WAS A 'SPECIAL' CALL
*
*  DETERMINING TYPE OF IBCD ARGUMENT
         TM 12(1),X'04'
         BC 12,ISIMP1
         MVI IBCD,X'84'      IS REAL*4 TYPE
         BC 15,ENTSEQ
ISIMP    SR 2,2              RESETTING THE
         ST 2,SPECFLG        'SPECIAL' FLAG
ISIMP1   MVI IBCD,X'82'      IS INTEGER*4 TYPE
         DROP 9
```

```
*  THE ENTRY SEQUENCE CODE
          USING REG11,11
          CNOP 0,4
ENTSEQ    STM 14,11,12(13)
          BAL 11,XENTSPEC(0,12)    BRANCH TO XENTSPEC RTN.
          DC H'0',CL6'SYMBOL'
          DC A(SAVE)
*  THE MODEL ARGUMENT LIST IS CREATED
          DC X'84',AL3(XPAGE)
          DC X'84',AL3(YPAGE)
          DC X'84',AL3(HEIGHT)
IBCD      DC X'00',AL(0) FILLED IN AT RUN-TIME
          DC X'84',AL3(ANGLE)
          DC X'80',AL3(NCHAR)
*  GENERATING THE OS-TYPE ARGUMENT LIST
          BAL 1,*+28
          DC X'00',AL3(XPAGE)
          DC X'00',AL3(YPAGE)
          DC X'00',AL3(HEIGHT)
DOPE      DC X'00',AL3(0)    FILLED IN AT RUN-TIME
          DC X'00',AL3(ANGLE)
          DC X'80',AL3(NCHAR)
          BALR 8,0
          USING *,8
          BC 15,SYMBOL2
```

```
*  AN ENTRY POINT USING OS-TYPE LINKAGE CONVENTIONS.
SYMB1    STM 14,12,12(13)    OS-TYPE ENTRY POINT
         LA 3,4              READJUSTING THE BASE
         SR 15,3             REGISTER
         LR 8,15             REG 15 MUST CONTAIN THE
         LA 9,SAVE           ADDRESS OF SYMB1
         ST 9,8(13)
         ST 13,SAVE+4
         LR 13,9
         LA 2,1              SETTING THE ENTRY TYPE
         ST 2,ENTFLG         FLAG TO ONE
SYMBOL2  LA 9,LINK           LOAD LINK FOR PLOT
```

SYMB1 represents an entry point which can be accessed by any calling routine using OS subprogram linkage conventions. The plotting routine NUMBER discussed next uses this entry point whenever it calls SYMBOL. A listing of SYMBOL appears in Appendix B.

NUMBER

The NUMBER routine converts the contents of a real variable or a real constant to an appropriately formatted number. The symbols used to represent the formatted number are plotted by calling the SYMBOL routine.

The calling statement is illustrated below.

```
CALL NUMBER(XPAGE,YPAGE,HEIGHT,FPN,ANGLE,NDEC)
```

The argument FPN represents the number that is to be converted and plotted. The argument NDEC controls the precision of the conversion. The other arguments are described in (7).

The NUMBER routine required very little modification. The OS type linkage code was replaced by the WATSL macro call

```
WATSL NUMBER,8,SAVE1,(84,84,84,84,84,82),10,
```

to generate WATFIV subprogram linkage code. In place of the call to SYMBOL, a call to the SYMB1 entry point was made. An Assembler listing of the code is found in Appendix B.

SCALE

The SCALE routine examines the REAL*4 data values passed down to it in a vector to determine a starting value and a scaling factor. This allows the range of the data values to be properly represented in the vector so that they fit in a given plotting area. The starting value is referred to as FIRSTV while the scaling value is referred to as DELTAV. Both these variables are referred to as the scaling parameters.

The calling sequence for SCALE is

```
CALL SCALE(ARRAY,AXLEN,NPTS,INC)
```

where

ARRAY is a vector containing the REAL*4 data values,

AXLEN is the length of the axis that the data is to be scaled to,

NPTS is the total number of data values in the array

INC is a number used as an increment in selecting data values in the vector.

The storing of FIRSTV and DELTAV in the unused elements of the vector presents a special problem for any modification of the SCALE routine for WATFIV use. The size of ARRAY must be at least two larger than the number of data values since FIRSTV is stored in the NPTS+1 location of ARRAY and DELTAV is stored in the location NPTS+2. If INC>1, then FIRSTV is stored in ARRAY(NPTS * INC + 1) and DELTAV is stored in ARRAY(NPTS * INC + INC + 1). The problem encountered is similiar to the one in SYMBOL regarding the IBCD argument when it is a REAL*4 or INTEGER*4 vector. WATFIV must have an extra argument passed to the subprogram to serve as the dimensioning variable for the ARRAY vector. Since SCALE was not written in Assembler code, the length of ARRAY cannot be obtained at run-time using the same technique used in SYMBOL.

The following technique was used to implement a WATFIV

version of SCALE that has the same argument list as the original version. An Assembler program called SCALE was written containing the four arguments ARRAY, AXLEN, NPTS, and INC. It contains the necessary Assembler instructions to obtain the length of ARRAY at run-time and move a copy of it to the subprogram's entry sequence code. This entry sequence code is present to check the subprogram linkage to ensure that the four arguments are passed down correctly. The name of the original SCALE subprogram is changed to SCALE1 and is always compiled by WATFIV. Since a WATFIV compiled subprogram contains its own entry sequence code, the modified SCALE routine sets up code for a WATFIV calling sequence to call SCALE1. An actual argument list is set up in SCALE which includes an extra argument to represent the dimensioning variable for ARRAY. An extra argument called NT is added to the dummy argument list of SCALE1. NT is the dimensioning variable for the ARRAY vector in the SCALE1 subprogram. The declaration statements for SCALE1 are illustrated below.

```
SUBROUTINE SCALE1(ARRAY,AXLEN,NN,INC,NT)
DIMENSION ARRAY(NT),SAVE(7)
```

A check is made in the SCALE1 subprogram to ensure that there is adequate storage space to store the contents of the scaling parameters. If the space isn't available, the

following error message is printed out before control returns to the calling program.

```
***ERROR***  THE VECTOR USED MUST BE DIMENSIONED LARGER
             THAN THE NUMBER OF DATA VALUES USED IN SCALE.
```

The SCALE and SCALE1 subprograms are listed in Appendix B.

AXIS

The AXIS routine is called to draw an axis line at any desired angle. It divides the line into 1 inch intervals with a number which has been appropriately scaled and drawn below the division marks. The axis line drawn is also labelled with an appropriate title.

The calling sequence for AXIS is shown below.

```
CALL AXIS(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,DELTAV)
```

The arguments are described in (7). The argument called IBCD requires the same special attention as it did in the SYMBOL routine. IBCD may be a simple variable (INTEGER*4 or REAL*4), a vector (INTEGER*4 or REAL*4), or a hollerith string. The approach taken to handle it in this routine is similiar to that taken in the SYMBOL routine. Appropriate code is set up to modify the entry sequence code at run-time according to the different calling argument types passed to

it from the calling programs.

The WATFIV version of the AXIS routine is implemented by creating an Assembler subprogram called AXIS whose function is to provide the appropriate entry sequence code to accommodate the different calling argument types. Upon successful execution of the entry sequence code in AXIS, the AXIS1 routine is called. The original version of the AXIS routine now compiled by WATFIV was renamed AXIS1. This subprogram required very little modification to run with WATFIV. The calling sequence used is the same one used for the Assembler subprogram AXIS with the exception that the dummy argument for IBCD is always treated as an INTEGER*4 vector containing only one element.

The purpose of the Assembler subprogram is to set up the appropriate entry sequence code for any allowed calling argument list and to pass on to AXIS1, the address of the first storage location of the set of alphameric characters to be plotted. The AXIS routine obtains this address after its entry sequence code is executed in the following way. If the actual argument for IBCD is a simple variable, the address of the storage location in AXIS is used for the corresponding call-by-value dummy argument. If the actual argument for IBCD is a vector, its starting address is found in the star routine for the dummy argument for IBCD. If the actual argument for IBCD is a hollerith string, the address

of the first character is in AXIS in the dope vector for the dummy vector for IBCD.

Once the address of the alphameric characters is obtained, it is passed on to the AXIS1 routine in the following way. A WATFIV calling sequence is set up to call AXIS1. The calling statement in FORTRAN would be

```
      CALL AXIS1(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,
     1FIRSTV,DELTAV)
```

where IBCD is an INTEGER*4 vector containing only one element. The following FORTRAN statements illustrate the code used in coding the AXIS1 subprogram.

```
      SUBROUTINE AXIS1(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,
     1FIRSTV,DELTAV)
      DIMENSION IBCD(1)
```

The address of the alphameric characters is stored in the star routine set up for the calling argument for IBCD in AXIS. When the call is made and the entry sequence code is executed in AXIS1, the prologue copies the address of the calling argument for IBCD into the star routine in AXIS1 for its dummy argument corresponding to IBCD.

The address in the star routine for IBCD in AXIS1 is passed on to the SYMBOL routine in the following way. The following FORTRAN statement calls the SYMBOL routine.

CALL SYMBOL(XT,YT,0.14,IBCD,ANGLE,KN)

The SYMBOL routine examines the calling argument list and appropriately modifies its entry sequence code to allow proper subprogram linkage to take place. WATFIV's prologue passes the address of the alphameric characters to the star routine set up in SYMBOL for its dummy argument corresponding to IBCD. SYMBOL then uses this address stored in this star routine as the address specifying the location of the first symbol to be plotted. SYMBOL ignores the rest of the information found in this star routine. IBCD was passed in the calling argument list for AXIS1 with a length of only one element just so that the star routine would be used to pass the address of the set of alphameric characters down to SYMBOL.

Errors may occur when calling SYMBOL when more characters are to be plotted than are available. The user must ensure that this does not happen. No error messages are given. SYMBOL will plot the desired number of symbols, but it plots a special symbol if it recognizes an invalid symbol occurring in the list of symbols to be plotted.

LINE

The LINE subprogram is used to draw a plotted line using pairs of data values passed to it in the vectors

XARRAY and YARRAY. These vectors contain data values representing the x and y coordinates of the data points. The scaling parameters, stored in these vectors, are used to scale the data values before drawing an actual plot, so that they will fit within the desired plotting range.

The calling sequence for LINE is as follows.

```
CALL LINE(XARRAY,YARRAY,NPTS,INC,LINTYP,INTEQ)
```

These arguments are described in (7).

The original version of the LINE subprogram can be correctly compiled by WATFIV but it will not execute properly. This is because of the convention concerning the method of storing the scaling parameters after the data values in the two vectors XARRAY and YARRAY. For proper WATFIV execution, a dimensioning variable for XARRAY and for YARRAY is required. These dimensioning variables do not occur in the argument list of the original version of the LINE subprogram. This variable dimensioning problem occurred before when modifying the SYMBOL, SCALE, and AXIS subprograms. The problem encountered with the LINE subprogram was resolved by writing an Assembler subprogram called LINE which is an interface between the original version of the LINE subprogram and the WATFIV calling program. The original version of the LINE subprogram was renamed LINE1. A WATFIV plotting program calls LINE which in

turn calls LINE1. The LINE1 subprogram contains the six arguments which are passed to LINE plus two additional arguments. The lengths of the XARRAY and YARRAY vectors are obtained at run-time in the Assembler subprogram LINE and are passed to the LINE1 subprogram as the two additional arguments. They are called DIM1 and DIM2. These dimensioning variables are used in the DIMENSION statement when declaring XARRAY and YARRAY in the LINE1 subprogram.

The subprogram declaration statements for LINE1 are illustrated below.

```
      SUBROUTINE LINE1(XARRAY,YARRAY,NPTS,INC,LINTYP,INTEQ,
     1DIM1,DIM2)
      DIMENSION XARRAY(DIM1),YARRAY(DIM2)
```

The rest of the subprogram required no further modifications. A listing of LINE and LINE1 appears in Appendix B.

SIMBLE AND AXISI

The modifications to SIMBLE and AXISI are very similiar to those performed on SYMBOL and AXIS. Listings of these routines appear in Appendix B.

Chapter 5

Batching of Plots

Since the FORTRAN G and H compilers do not allow the user to batch plots together, the plotter tape only contains one plot. Batching of plots has the advantage that more than one plot may be stored on a plotter tape. This approach is much more efficient and can greatly improve turn around time for the user who has several plot jobs to run. WATFIV does allow batching of programs and the Calccmp plot routines have been modified to accommodate this type of processing.

There is one problem using the Calcomp Plot routines in this kind of environment. The possibility exists that one may accidently or intentionally write on another user's plot. The user has control over the pen position when he closes the tape. When there is more than one plot written on a plotter tape, there is no operator assistance to set the paper between different plots. The closing pen position of one plot will also be the starting pen position for the next plot. It is important that the user leaves the pen in an appropriate position before closing the tape or he may plot on someone else's plot or have his own plot results overwritten.

The problem has been approached in the following way. Each plot is associated with a plot region on the paper in

which it is drawn. The starting pen position is the leftmost point of the plot region. The rightmost x coordinate position is taken to represent the rightmost point of the plot region. The user should position the pen beyond this point before or as he closes the tape.

The PLOT routine keeps track of this plot region through the use of several variables. The leftmost x point is set at zero initially and no x coordinate point is allowed to be plotted to the left of it. The rightmost x point is updated appropriately to represent the rightmost x point plotted for a given plot. Since the size of a plot is actually measured in plotter steps (ie. 1/100 of an inch), all these variables keep track of the number of plotter steps used by a plot. Every time a new origin point is requested, the variables are updated to reflect the change.

When the last call to PLOT is detected, a check is made to ensure that the closing x position of the pen is beyond the rightmost x point plotted so far. If it isn't, the rightmost point of the plot region is extended to the right by 2 inches and a warning flag is set. If an attempt was made to plot left of the leftmost point of the plot region, an error message flag is set. Control immediately returns to the calling program. These warning and error messages are printed out from the WATFIV subprogram called ERROR1. These messages are printed out just before the tape is closed.

The warning message is illustrated below.

WARNING CLOSING X COORDINATE POSITION SHOUD BE BEYOND RIGHTMOST X POINT PLOTTED.

The error message is illustrated below.

ERROR ATTEMPT TO MOVE LEFT FURTHER THAN PLOT REGION ALLOWS. MOVE PLOT ORIGIN(S) AN APPROPRIATE NUMBER OF POSITIONS TO THE RIGHT.

Chapter 6

Conclusions

In order to illustrate that the modified Calcomp plot routines produce the same graphic plots as the original Calcomp plot routines, the sample program illustrated in (7) was run using both sets of routines. A listing of these FORTRAN programs is found in Appendix C along with the graphic plot produced. The plot results produced were identical in both cases. The object of this thesis was accomplished.

Graphical plotting routines using the FORTRAN G and H compilers are available at the University of Manitoba. At present, most students are taught programming and programming techniques through the use of FORTRAN IV with the WATFIV compiler. A set of graphical plot routines described in (7) are presently supported by WATFIV at this university but these are not compatible with the Calcomp plot routines. The University of Waterloo also has some Calcomp plotting routines which run under WATFIV (10). Howrever, these routines are not totally compatible with the basic set of Calcomp plot routines which run under FORTRAN G and H. This thesis was motivated by the desire to allow students access to the more universally used Calcomp plot routines using the WATFIV compiler with which they are more

familiar.

If entry points are provided at a later date for the XENTSPEC, XA1, and XRET routines, the referencing techniques used would only require slight modifications. A request of this nature will be forwarded to the WATFIV compiler people who are presently working on updating new versions of the compiler at Waterloo, Ontario.

We feel a logical extension to this thesis is to provide an environment independent approach to graphical plotting. At present, a graphical plot can only be displayed by drawing it on paper with the Calcomp plotter. Graphics facilities could be provided to display a graphical plot on an on-line graphics display terminal. Users would be able to see their plot before requesting a drawing of it made. The Calcomp plotter would only be used to draw final versions of plots and would not have to be used as the plot is being developed.

APPENDIX A

This appendix contains a program listing of the WATSL macro.

T CODE ADDR1 ADDR2 STMT SOURCE STATEMENT 20 SEP 73

```
                1          MACRO
                2 &TP      WATSL &NAME,&BASE,&SAVE,&ARGLIST,&RTNTYPE,&ARRAY
                3 .*
                4 .*********************************************************************
                5 .*
                6 .*  THIS MACRO GENERATES WATFIV ENTRY SEQUENCE CODE IN AN ASSEMBLER
                7 .*  PROGRAM.  IT CAN HANDLE CONSTANTS, SIMPLE VARIABLES, ONE
                8 .*  DIMENSIONAL ARRAYS, AND A HOLLERITH STRING AS SUBPROGRAM
                9 .*  ARGUMENTS.
               10 .*
               11 .*  &NAME     - NAME OF THE SUBPROGRAM
               12 .*  &BASE     - BASE REGISTER TO BE USED, MAY BE LEFT BLANK
               13 .*  &SAVE     - NAME OF THE SAVEAREA
               14 .*  &ARGLIST - LIST OF PARAMETERS CODE IS TO BE GENERATED FOR.
               15 .*  &RTNTYPE - TYPE OF THE ROUTINE
               16 .*                  10 - SUBROUTINE
               17 .*                  20 - FUNCTION
               18 .*  &APRAY    - USED AS A FLAG TO CAUSE SPECIAL CODE TO BE GENERATED.
               19 .*
               20 .*********************************************************************
               21 .*
               22          GBLA &GBBE
               23          LCLA &TEMP,&LABEL,&STPTR,&KTNVEC,&TEMP1,&DUMMY
               24          LCLA &APTR(20),&TEMP2
               25          LCLA  &DOPE(30),&KTNDOP
               26          LCLC &ADDRESS(30),&LOCATIO(30),&TEMPC,&TEMPC1
               27          LCLC &LOC,&STAR,&LST,&DOP
               28 .*
               29 .*********************************************************************
               30 .*
               31 .*   SETTING THE TEMPORARIES AND COUNTERS.
               32 .*
               33 .*********************************************************************
               34 .*
               35 &GBBE    SETA &GBBE+1
               36 &TEMP    SETA &GBBE
               37 &LOC     SETC 'LOC&GBBE'
               38 &STAR    SETC 'STAR&GBBE'
               39 &LST     SETC 'LST&GBBE'
               40 &DOP     SETC 'DOP&GBBE'
               41 .*
               42 .*********************************************************************
               43 .*
               44 .*  THE WATFIV ENTRY SEQUENCE CODE IS BEING GENERATED.
               45 .*
               46 .*********************************************************************
               47 .*
               48          USING R11&TEMP,11
               49          CNOP 0,4
               50 &TP      STM 14,11,12(13)        13 CONTAINS CALLING SAVE AREA'S ADDRESS
               51          BAL 11,XENTSPEC(0,12)   BRANCH TO XENTSPEC  ROUTINE
               52 R11&TEMP DC H'0',CL6'&NAME'
               53          DC A(SAVE)              ADDRESS OF A 24 WORD SAVE AREA
               54 .*
               55 .*********************************************************************
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                        56 .*
                        57 *   THE MODEL ARGUMENT LIST IS CREATED
                        58 .*
                        59 .*****************************************************************
                        60 .*
                        61 .AGAIN   ANOP
                        62 &LABEL   SETA &LABEL+1
                        63          AIF (&LABEL GT N'&ARGLIST).EN
                        64 &TEMP    SETA &ARGLIST(&LABEL)                       IT WAS
                        65          AIF (&TEMP GT 200).VARE                     AN INDICATOR CODE
                        66          AIF (&TEMP EQ 82 OR &TEMP EQ 84).VAR        A SIMPLE VARIABLE
                        67          AIF (&TEMP EQ 92 OR &TEMP EQ 94).VECTOR     A VECTOR
                        68          AIF (&TEMP EQ 9).EAP                        A HOLLERITH STRING
                        69          MNOTE 'INVALID TYPE FOUND - NO CODE GENERATED FOR &TEMP'
                        70          AGO .AGAIN
                        71 .*
                        72 .*****************************************************************
                        73 .*
                        74 .*  CODE IS GENERATED FOR A HOLLERITH STRING
                        75 .*
                        76 .*****************************************************************
                        77 .*
                        78 .EAP     ANOP
                        79 &KTNDOP  SETA &KTNDOP+1
                        80          DC X'09',AL3(&DOP&KTNDOP)     ADDRESS OF DOPE VECTOR
                        81 &STPTR   SETA &STPTR+1
                        82 &ADDRESS(&STPTR) SETC '&DOP&KTNDOP'    TO BE LATER CHANGED AT RUN-TIME
                        83 &DOPE(&KTNDOP) SETA &STPTR
                        84          AGO .AGAIN
                        85 .*
                        86 .*****************************************************************
                        87 .*
                        88 .*  AN INTEGER USED AS A VARIABLE DIMENSION ELEMENT IS FOUND.
                        89 .*  ADDITIONAL PROCESSING IS PERFORMED TO KEEP TRACK OF THE ARRAY
                        90 .*  NAME THAT IS TO BE ASSOCIATED WITH THIS VARIABLE DIMENSION
                        91 .*  ELEMENT.
                        92 .*
                        93 .*****************************************************************
                        94 .*
                        95 .VARE    ANOP
                        96 &TEMP1   SETA 1
                        97 .AEPO    ANOP
                        98 &TEMP2   SETA &ARGLIST(&LABEL+&TEMP1)
                        99          AIF (&TEMP2 LE 200).EEE
                       100 &TEMP1   SETA &TEMP1+1
                       101          AGO .AEPO
                       102 .EEE     ANOP
                       103 &DUMMY   SETA &DUMMY+&TEMP1
                       104 .*
                       105 .*****************************************************************
                       106 .*
                       107 .*  LABEL NOW POINTS TO THE ARGUMENT TYPE.
                       108 .*
                       109 .*****************************************************************
                       110 .*
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                       111 &TEMP2   SETA &LABEL
                       112 &LABEL   SETA &LABEL+&TEMP1
                       113 .AEET    ANOP
                       114 &TEMP    SETA &TEMP-200
                       115 &TEMP1   SETA &APTR(&TEMP)
                       116 &LOCATIO(&TEMP1) SETC '&LOC&LABEL'
                       117 &TEMP2   SETA &TEMP2+1
                       118 &TEMP    SETA &ARGLIST(&TEMP2)
                       119          AIF (&TEMP2 LT &LABEL).AEET
                       120 .*
                       121 .*********************************************************************
                       122 .*
                       123 .*  CODE IS GENERATED FOR A SIMPLE VARIABLE
                       124 .*
                       125 .*********************************************************************
                       126 .*
                       127 .VAR     DC X'&TEMP',AL3(&LOC&LABEL)
                       128 &STPTR   SETA &STPTR+1
                       129 &ADDRESS(&STPTR) SETC '&LOC&LABEL'
                       130          AGO .AGAIN
                       131 .*
                       132 .*********************************************************************
                       133 .*
                       134 .*  CODE IS GENERATED FOR AN ARRAY NAME
                       135 .*
                       136 .*********************************************************************
                       137 .*
                       138 .VECTOR  ANOP
                       139 &KTNVEC  SETA &KTNVEC+1
                       140 &STPTR   SETA &STPTR+1
                       141 &APTR(&KTNVEC) SETA &STPTR
                       142 &ADDRESS(&STPTR) SETC '&LST&KTNVEC'
                       143          DC X'&TEMP',AL3(&STAR&KTNVEC)
                       144          AGO .AGAIN
                       145 .EN      DC X'&RTNTYPE',AL3(0)
                       146 .*
                       147 .*********************************************************************
                       148 .*
                       149 .*  THE MODEL ARGUMENT LIST HAS BEEN SET UP
                       150 .*
                       151 .*********************************************************************
                       152 .*
                       153          AIF (&KTNVEC EQ 0).REGLIST
                       154          AIF (N'&ARRAY EQ 0).FLA
                       155 .*
                       156 .*********************************************************************
                       157 .*
                       158 .*  GENERATING A TEMPORARY STORAGE LOCATION TO BE USED BY A STAR RTN.
                       159 .*
                       160 .*********************************************************************
                       161 .*
                       162          BC 15,*+8
                       163 TPST&GBBE DC F'0'
                       164 .FLA     ANOP
                       165 .*
```

```
CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                      166 .*********************************************************************
                      167 .*
                      168 *
                      169 *    GENERATING CODE FOR THE STAR ROUTINES FOR THE ARRAY NAMES
                      170 *
                      171 .*
                      172 .*********************************************************************
                      173 .*
                      174 &TEMP1    SETA 1
                      175 .TR2      BC 15,*+32
                      176          CNOP 2,4
                      177          DC CL6' ' THE ARRAY NAME IS UNKNOWN AT THIS TIME
                      178 &STAR&TEMP1 BAL 15,XA1(0,12)  BRANCH TO XA1
                      179 &TEMP     SETA &APTR(&TEMP1)
                      180 *    THE NEXT THREE ADDRESSES ARE CREATED AT RUN-TIME BY THE WATFIV
                      181 *                      PROLOGUE.
                      182 &ADDRESS(&TEMP) DC AL1(0),AL3(0)   LST__ IS THE ADDRESS OF THE ARRAY
                      183          DC AL1(2),AL3(0)
                      184          DC A(0)
                      185          AIF (N'&ARRAY EQ 0).AD1
                      186          DC X'40',AL3(TPST&GBBE)
                      187          AGO .F11D
                      188 .AD1     DC X'40',AL3(&LOCATIO(&TEMP))
                      189 .F11D    ANOP
                      190 &TEMP1   SETA &TEMP1+1
                      191          AIF (&TEMP1 LE &KTNVEC).TR2
                      192 .*
                      193 .*********************************************************************
                      194 .*
                      195 .*  FINISHED GENERATING STAR ROUTINES
                      196 .*
                      197 .*********************************************************************
                      198 .*
                      199 .REGLIST  ANOP
                      200          AIF (N'&ARGLIST EQ 0).TP2
                      201 .*
                      202 .*********************************************************************
                      203 .*
                      204 *
                      205 *    GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                      206 *
                      207 .*
                      208 .*********************************************************************
                      209 .*
                      210 &LABEL    SETA &LABEL-&DUMMY
                      211 ABL&GBBE BAL 1,*+4*&LABEL       AN IN-LINE ARGUMENT LIST IS USED.
                      212 &TEMP     SETA 1
                      213 .TP1      AIF (&TEMP GE &LABEL-1).EP3
                      214 &TEMPC    SETC '&ADDRESS(&TEMP)'
                      215          AIF ('&TEMPC'(1,3) EQ 'DOP').EP4
                      216          AIF ('&TEMPC'(1,3) EQ 'LST').GAB
                      217          DC X'00',AL3(&ADDRESS(&TEMP))
                      218          AGO .GAB1
                      219 .GAB     DC X'00',AL3(0)      FILLED IN LATER BY A MVC INSTRUCTION.
                      220          AGO .GAB1
```

```
                    221 .EP4      ANOP
                    222 &TEMPC    DC X'00',AL3(0)      DOPE VECTOR - FILLED IN AT RUN TIME
                    223 .GAB1     ANOP
                    224 &TEMP     SETA &TEMP+1
                    225           AGO .TP1
                    226 .EP3      DC X'80',AL3(&ADDRESS(&TEMP))
                    227 .*
                    228 .*********************************************************************
                    229 .*
                    230 .*  THE ARGUMENT LIST HAS BEEN CREATED ONCE,
                    231 .*  HOWEVER THE ARRAY ADDRESSES ARE INCORRECT.
                    232 .*  THE FOLLOWING CODE MOVES THE ARRAY ADDRESSES
                    233 .*  TO THE APPROPRIATE PLACES IN THE ARGUMENT LIST.
                    234 .*
                    235 .*********************************************************************
                    236 .*
                    237           AIF (&KTNVEC EQ 0).ACCF
                    238 *
                    239 *    A MVC INSTRUCTION MOVES AN ARRAY ADDRESS INTO THE OS-TYPE ARGUMENT
                    240 *    LIST AT RUN-TIME.
                    241 *
                    242 &TEMP     SETA 1
                    243 &TEMPC1 SETC 'ABL&GBBE'
                    244 .AEEP     ANOP
                    245 &TEMPC    SETC '&ADDRESS(&TEMP)'
                    246           AIF ('&TEMPC'(1,3) NE 'LST').ABB
                    247           MVC &TEMPC1+4*&TEMP+1(3),&TEMPC+1
                    248 .ABB      ANOP
                    249 &TEMP     SETA &TEMP+1
                    250           AIF (&TEMP LT &LABEL).AEEP
                    251 .*
                    252 .*********************************************************************
                    253 .*
                    254 .*  FINISHED MOVING ADDRESSES AROUND
                    255 .*
                    256 .*********************************************************************
                    257 .*
                    258           CNOP 0,4
                    259 .ACCF     ANOP
                    260 .*
                    261 .*********************************************************************
                    262 .*
                    263 .*  THE ARGUMENT LIST HAS NOW BEEN COMPLETED
                    264 *
                    265 *    GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                    266 *
                    267 .*********************************************************************
                    268 .*
                    269 &TEMP     SETA 1
                    270           BC 15,*+4*(&LABEL-&KTNVEC-&KTNDOP)
                    271 .ADD      ANOP
                    272 &TEMPC    SETC '&ADDRESS(&TEMP)'
                    273           AIF ('&TEMPC'(1,3) EQ 'LST').AGB
                    274           AIF ('&TEMPC'(1,3) EQ 'DOP').AGB
                    275 &TEMPC    DS F
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                       276 .AGB      ANOP
                       277 &TEMP     SETA &TEMP+1
                       278           AIF (&TEMP LT &LABEL).ADO
                       279 .*
                       280 .*********************************************************************
                       281 .*
                       282 .*  THE STORAGE LOCATIONS HAVE BEEN GENERATED
                       283 .*
                       284 .*********************************************************************
                       285 .*
                       286 .*  CHECKING TO SEE IF CODE FOR A BASE REGISTER ASSIGNMENT HAS TO BE
                       287 .*            GENERATED.
                       288 .*
                       289 .*********************************************************************
                       290 .*
                       291 .TP2      ANOP
                       292           AIF (N'&BASE EQ 0).END
                       293           DROP 11
                       294 *
                       295 *
                       296           BALR &BASE,0
                       297           USING *,&BASE
                       298 .END      MEND
```

```
:T CODE   ADDR1 ADDR2  STMT   SOURCE STATEMENT

                       300 PLOTS    START 0    PRODUCT NO. 18600   VERSION V097  DATE  SEPT 1967
                       301 *          COPYRIGHT 1967  CALIFORNIA COMPUTER PRODUCTS INC.          PLOT0020
                       302 * FOR IBM SYSTEM 360  OS      CALCOMP  750/500 SYSTEM  800 BPI  9-TRACK PLOT0030
                       303 *
                       304 *   THESE CALCOMP PLOT ROUTINES HAVE BEEN MODIFIED AT THE UNIVERSITY
                       305 *   OF MANITOBA TO RUN WITH WATFIV PROGRAMS.
                       306 *                                                                      PLOT0040
                       307 *                                                                      PLOT0050
                       308 ************************************************************************* PLOT0060
                       309 *                                                                      * PLOT0070
                       310 *        FORTRAN LINKAGE FOR PLOT ROUTINE                              * PLOT0080
                       311 *                                                                        PLOT0090
                       312 *        ENTRY PLOTS                                                   * PLOT0100
                       313 *     CALL PLOTS(IBUF,NLOC,LDEV)                                       * PLOT0110
                       314 *             IBUF IS THE FIRST LOCATION IN A WORK AREA                * PLOT0120
                       315 *             NLOC IS THE NUMBER OF WORDS IN THE WORK AREA             * PLOT0130
                       316 *             LDEV IS NOT USED IN THIS SYSTEM                          * PLOT0160
                       317 *                                                                        PLOT0170
                       318           ENTRY PLOT          WATFIV TYPE ENTRY POINT
                       319           ENTRY PLOT1         OS-TYPE ENTRY POINT
                       320           EXTRN ERROR1        WATFIV ERROR MESSAGE ROUTINE.
                       321 *     CALL PLOT(XPAGE,YPAGE,IPEN)                                      * PLOT0180
                       322 *             (XPAGE,YPAGE) IS THE PLOTTER PAGE COORDINATES IN INCHES* PLOT0190
                       323 *             IPEN IS THE PEN INDICATOR (LEGAL VALUES ARE              * PLOT0200
                       324 *                  2,3,12,13,22,23,999,-2,-3,-12,-13)                  * PLOT0210
                       325 *                                                                      * PLOT0220
                       326 *                                                                        PLOT0230
                       327 *        ENTRY FACTOR                                                  * PLOT0240
                       328 *     CALL FACTOR(FACT)                                                * PLOT0250
                       329 *             FACT IS A MULTIPLICATIVE FACTOR FOR ENTIRE PLOT          * PLOT0260
                       330 *                                                                        PLOT0270
                       331 *        ENTRY WHERE                                                   * PLOT0280
                       332 *     CALL WHERE(RXPAGE,RYPAGE,RFACT)                                  * PLOT0290
                       333 *             (RXPAGE,RYPAGE) IS THE CURRENT PEN LOCATION              * PLOT0300
                       334 *             RFACT IS THE CURRENT FACTOR                              * PLOT0310
                       335 *                                                                        PLOT0320
                       336 *        ENTRY OFFSET                                                  * PLOT0330
                       337 *     CALL OFFSET(XOFF,YFCT,YOFF,YFCT)                                 * PLOT0340
                       338 *             XOFF IS THE XPAGE OFFSET                                 * PLOT0350
                       339 *             XFCT IS THE XPAGE FACTOR                                 * PLOT0360
                       340 *             YOFF IS THE YPAGE OFFSET                                 * PLOT0370
                       341 *             YFCT IS THE YPAGE FACTOR                                 * PLOT0380
                       342 ************************************************************************* PLOT0390
                       343 *
                       344 *
                       345 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                       346 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                       347 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                       348 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                       349 *
                       350 XENTSPEC EQU 246
                       351 XRET     EQU 1124
                       352 XA1      EQU 1592
                       353 *
                       354 *
```

```
CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                      355 *
                      356 *
                      357          WATSL PLOTS,,SAVE,(92,201,82),10,
                      358+         USING R111,11
                      359+         CNOP  0,4
00C             0000C 360+         STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
0F6             000F6 361+         BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
03D6E3E240             362+R111    DC    H'0',CL6'PLOTS'
E4                     363+         DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                       364+*   THE MODEL ARGUMENT LIST IS CREATED
2C                     365+         DC    X'92',AL3(STAR11)
58                     366+         DC    X'82',AL3(LOC13)
00                     367+         DC    X'10',AL3(0)
                       368+*
                       369+*   GENERATING CODE FOR THE STAR ROUTINES FOR THE ARRAY NAMES
                       370+*
038             C0040  371+         BC    15,*+32
                       372+         CNOP  2,4
404040                 373+         DC    CL6' ' THE ARRAY NAME IS UNKNOWN AT THIS TIME
638             00638  374+STAR11   BAL   15,XA1(0,12) BRANCH TO XA1
                       375+*   THE NEXT THREE ADDRESSES ARE CREATED AT RUN-TIME BY THE WATFIV
                       376+*                 PROLOGUE.
00                     377+LST11    DC    AL1(0),AL3(0) LST__ IS THE ADDRESS OF THE ARRAY
00                     378+         DC    AL1(2),AL3(0)
00                     379+         DC    A(0)
58                     380+         DC    X'40',AL3(LOC13)
                       381+*
                       382+*   GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                       383+*
044             0004C  384+ABL1     BAL   1,*+4*3 AN IN-LINE ARGUMENT LIST IS USED.
00                     385+         DC    X'00',AL3(0) FILLED IN LATER BY A MVC INSTRUCTION.
58                     386+         DC    X'80',AL3(LOC13)
                       387+*
                       388+*   A MVC INSTRUCTION MOVES AN ARRAY ADDRESS INTO THE OS-TYPE ARGUMENT
                       389+*   LIST AT RUN-TIME.
                       390+*
03D D029 00045 00031   391+         MVC   ABL1+4*1+1(3),LST11+1
                       392+         CNOP  0,4
                       393+*
                       394+*   GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                       395+*
054             00C5C  396+         BC    15,*+4*(3-1-0)
                       397+LOC13    DS    F

                       399 AXFZ     BCR 0,0           TO BE MODIFIED TO PREVENT REENTRY
                       400          BCR 0,0
                       401          CNOP 0,4
                       402 *  THIS INSTRUCTION MOVES CODE TO ALLOW ENTRY TO THE OS-TYPE ENTRY TO
                       403 *                PLOT
24E D054 00256 00C5C   404          MVC PLOT1(2),AXFZ
246 D054 0024E 00C5C   405          MVC ALIZ(4),AXFZ  ALLOWS EXECUTION OF THE WATFIV ENTRY TO PLOT
054 9540 00C5C 00548   406          MVC AXFZ(4),AGBZ  MODIFIES CODE TO PREVENT REENTRY TO PLOTS
                       407 *  FIX LXP AT - .53
035             00035  408          LA 3,53
                       409          LNR 3,3
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    PLOT0480
 B54C             00554   410           ST 3,LXP                                                     PLOT0490
 BA0C             00A14   411           LE    0,STPSZ                                                PLOT0500
 B5A0             005A8   412           ME    0,FACT                                                 PLOT0530
 B59C             005A4   413           STE   0,FACSZ             FACSZ=FACT*STPSZ                   PLOT0540
                          414           SR    0,0                                                    PLOT0550
 B574             0057C   415           ST    0,NPX               NPX=0                              PLOT0560
 B578             00580   416           ST    0,NPY               NPY=0                              PLOT0570
 B57C             00584   417           ST    0,ICOLD             ICOLD=0                            PLOT0580
 B580             00588   418           ST    0,XOLD              XOLD=0                             PLOT0590
 B584             0058C   419           ST    0,YOLD              YOLD=0                             PLOT0600
 1000             C0000   420           LM    2,4,0(1)            LOAD LINKAGE                       PLOT0610
 B5CC             005D4   421           N     2,ADDMSK       REMOVE ANY HIGH ORDER BITS              PLOT0620
                          422           LTR   3,3                 TEST FOR THREE ARGUMENTS           PLOT0630
 B09A             000C2   423           BC    4,NODEV              IF THREE,
 4000             C0000   424           L     4,0(0,4)              PICK UP THIRD ARGUMENT
 B5DE             005E6   425           CH    4,N999              CHECK IF RESTRICTIONS EASED
 B09A             000C2   426           BNE   NODEV               RESTRICTIONS
 B5E0 B5E8 005E8  005F0   427           MVC   CHECKX,XMAX         EASE RESTRICTIONS
 B5D8             005E0   428           STH   4,LGLDEV            STORE AS LOGICAL DEVICE            PLOT0640
 B246             0024E   429 NODEV     LA 10,ALIZ                SET UP NEW BASE REGISTER
                          430           DROP 11

                          432           USING ALIZ,10
 3000             00000   433           L     3,0(0,3)            PICK UP COUNT                      PLOT0670
 A5B4             00802   434           BC    15,OPTAP            BRANCH TO OPEN TAPE                PLOT0680
                          435           DROP 10

                          437 *
                          438 ********************************************************************************
                          439 *                                                                              *
                          440 *          FACTOR                                                              *
                          441 *                                                                              *
                          442 ********************************************************************************
                          443 *
                          444 FACTOR    WATSL FACTOR,9,SAVE,(84),10,
                          445+          USING R112,11
                          446+          CNOP  0,4
 D00C             0000C   447+FACTOR    STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
 C0F6             000F6   448+          BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
C6C1C3E3D6D9             449+R112      DC    H'0',CL6'FACTOR'
04E4                     450+          DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                         451+*   THE MODEL ARGUMENT LIST IS CREATED
00F8                     452+          DC    X'84',AL3(LOC21)
0000                     453+          DC    X'10',AL3(0)
                         454+*
                         455+*   GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                         456+*
 B01C             000F4   457+ABL2      BAL   1,*+4*2 AN IN-LINE ARGUMENT LIST IS USED.
00F8                     458+          DC    X'80',AL3(LOC21)
                         459+*
                         460+*   GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                         461+*
 B024             000FC   462+          BC    15,*+4*(2-0-0)
                         463+LOC21     DS    F
                         464+          DROP  11
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    PLOT0730
                       465+*
                       466+*
                       467+          BALR  9,0
                       468+          USING *,9
0000              00000  470           L     2,0(1)          LOAD LINKAGE                         PLOT0730
2000              00000  471           LE    0,0(0,2)        PICK UP FACTOR                       PLOT0740
94AA              005A8  472           LE    2,FACT                                               PLOT0750
                         473           DER   2,0                                                  PLOT0760
                         474           LER   4,2                                                  PLOT0770
948A              00588  475           ME    2,XOLD                                               PLOT0780
948A              00588  476           STE   2,XOLD     XOLD=XOLD*(OLD FACTOR/NEW FACTOR)         PLOT0790
948E              0058C  477           ME    4,YOLD                                               PLOT0800
948E              0058C  478           STE   4,YOLD     YOLD=YOLD*(OLD FACTOR/NEW FACTOR)         PLOT0810
94AA              005A8  479           STE   0,FACT          STORE FACT                           PLOT0820
9916              00A14  480           ME    0,STPSZ                                              PLOT0830
94A6              005A4  481           STE   0,FACSZ         FACSZ=FACT*STPSZ                     PLOT0840
                         482 *    A  WATFIV  TYPE  RETURN
C464              00464  483           BC 15,XRET(0,12)      BRANCH TO XRET RTN.
                         484           DROP 9

                         486 *
                         487 ************************************************************************
                         488 *                                                                      *
                         489 *             WHERE                                                    *
                         490 *                                                                      *
                         491 ************************************************************************
                         492 *
                         493 WHERE     WATSL WHERE,9,SAVE,(84,84,84),10,
                         494+          USING R113,11
                         495+          CNOP  0,4
D00C              0000C  496+WHERE     STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
C0F6              000F6  497+          BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
6C8C5D9C540               498+R113     DC    H'0',CL6'WHERE'
454                       499+         DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                         500+*    THE MODEL ARGUMENT LIST IS CREATED
168                      501+         DC    X'84',AL3(LOC31)
16C                      502+         DC    X'84',AL3(LOC32)
170                      503+         DC    X'84',AL3(LOC33)
000                      504+         DC    X'10',AL3(0)
                         505+*
                         506+*    GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                         507+*
B02C              00164  508+ABL3      BAL   1,*+4*4 AN IN-LINE ARGUMENT LIST IS USED.
168                      509+         DC    X'00',AL3(LOC31)
16C                      510+         DC    X'00',AL3(LOC32)
170                      511+         DC    X'80',AL3(LOC33)
                         512+*
                         513+*    GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                         514+*
B03C              00174  515+          BC    15,*+4*(4-0-0)
                         516+LOC31     DS    F
                         517+LOC32     DS    F
                         518+LOC33     DS    F
                         519+          DROP  11
```

```
 CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                       520+*
                       521+*
                       522+          BALR  9,0
                       523+          USING *,9
1000                   524           LM    2,4,0(1)        LOAD LINKAGE                          PLOT0910
                       525           LTR   3,3             TEST FOR THREE ARGUMENTS              PLOT0920
?012                   526           BC    4,NOFACT        IF THREE,                             PLOT0930
?432                   527           L     5,FACT           PICK UP FACT AND                     PLOT0940
?C00                   528           ST    5,0(0,4)         STORE AS THIRD ARGUMENT              PLOT0950
?412                   529 NOFACT    LM    4,5,XOLD        PICK UP XOLD AND YOLD                 PLOT0960
?000                   530           ST    4,0(0,2)        STORE XOLD AS FIRST ARGUMENT          PLOT0970
?000                   531           ST    5,0(0,3)        STORE YOLD AS SECOND ARGUMENT         PLOT0980
                       532 *  A WATFIV TYPE RETURN
?464                   533           BC 15,XRET(0,12)      BRANCH TO XRET RTN.
                       534           DROP 9

                       536 *
                       537 ************************************************************************
                       538 *                                                                      *
                       539 *            OFFSET                                                    *
                       540 *                                                                      *
                       541 ************************************************************************
                       542 *
                       543 OFFSET    WATSL OFFSET,9,SAVE,(84,84,84,84),10,
                       544+          USING R114,11
                       545+          CNOP  0,4
?00C                   546+OFFSET    STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
?0F6                   547+          BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
?C6C6E2C5E3            548+R114      DC    H'0',CL6'OFFSET'
?E4                    549+          DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                       550+*   THE MODEL ARGUMENT LIST IS CREATED
?D8                    551+          DC    X'84',AL3(LOC41)
?DC                    552+          DC    X'84',AL3(LOC42)
?E0                    553+          DC    X'84',AL3(LOC43)
?F4                    554+          DC    X'84',AL3(LOC44)
?00                    555+          DC    X'10',AL3(0)
                       556+*
                       557+*   GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                       558+*
?034                   559+ABL4      BAL   1,*+4*5 AN IN-LINE ARGUMENT LIST IS USED.
?08                    560+          DC    X'00',AL3(LOC41)
?DC                    561+          DC    X'00',AL3(LOC42)
?E0                    562+          DC    X'00',AL3(LOC43)
?E4                    563+          DC    X'80',AL3(LOC44)
                       564+*
                       565+*   GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                       566+*
?048                   567+          BC    15,*+4*(5-0-0)
                       568+LOC41     DS    F
                       569+LOC42     DS    F
                       570+LOC43     DS    F
                       571+LOC44     DS    F
                       572+          DROP  11
                       573+*
                       574+*
```

```
 CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                       575+          BALR  9,0
                       576+          USING *,9
000            00000   577           LM    2,5,0(1)      *  LOAD LINKAGE                    PLOT1050
000            00000   578           L     2,0(0,2)      *  PICK UP XOFF                    PLOT1060
000            00000   579           L     3,0(0,3)      *          XFCT                    PLOT1070
000            00000   580           L     4,0(0,4)      *          YOFF                    PLOT1080
000            00000   581           L     5,0(0,5)      *          YFCT                    PLOT1090
0D2            002BC   582           STM   2,5,XOFF      *  STORE XOFF,XFCT,YOFF,YFCT       PLOT1100
464            00464   583           BC 15,XRET(0,12)  BRANCH TO XRET RTN.
                       584           DROP 9


                       587 *
                       588 ************************************************************************
                       589 *                                                                      *
                       590 *           PLOT                                                       *
                       591 *                                                                      *
                       592 ************************************************************************
                       593 *
                       594 PLOT      WATSL PLOT,10,SAVE,(84,84,82),10,
                       595+          USING R115,11
                       596+          CNOP  0,4
00C            0000C   597+PLOT      STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
0F6            000F6   598+          BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
7D3D6E34040            599+R115      DC    H'0',CL6'PLOT'
E4                     600+          DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                       601+*   THE MODEL ARGUMENT LIST IS CREATED
240                    602+          DC    X'84',AL3(LOC51)
244                    603+          DC    X'84',AL3(LOC52)
248                    604+          DC    X'82',AL3(LOC53)
000                    605+          DC    X'10',AL3(0)
                       606+*
                       607+*   GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                       608+*
02C            0023C   609+A8L5      BAL   1,*+4*4 AN IN-LINE ARGUMENT LIST IS USED.
240                    610+          DC    X'00',AL3(LOC51)
244                    611+          DC    X'00',AL3(LOC52)
248                    612+          DC    X'80',AL3(LOC53)
                       613+*
                       614+*   GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                       615+*
03C            0024C   616+          BC    15,*+4*(4-0-0)
                       617+LOC51     DS    F
                       618+LOC52     DS    F
                       619+LOC53     DS    F
                       620+          DROP  11
                       621+*
                       622+*
                       623+          BALR  10,0
                       624+          USING *,10
                       625 *  MODIFIED BY PLOTS TO ALLOW ENTRY TO PLOT.
464            00464   626 ALIZ      BC 15,XRET(0,12)      BRANCH TO XRET ROUTINE
02C            0027A   627           BC 15,PLOT2           BRANCH AROUND THE OS-TYPE ENTRY POINT.
                       628 *
                       629 *  CODE  FOR  AN  OS-TYPE  ENTRY  POINT
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                       630 *
                       631 PLOT1    BR 14                MODIFIED BY THE PLOT ROUTINE
D00C           0000C   632          STM 14,12,12(13)
C008           00008   633          LA 3,8
                       634          SR 15,3              REGISTER 15 MUST ALWAYS POINT TO PLOT1
                       635          LR 10,15
A296           004E4   636          LA 9,SAVE            USING OS-TYPE SAVE CONVENTIONS
0008           00008   637          ST 9,8(13)
A29A           004E8   638          ST 13,SAVE+4
                       639          LR 13,9
0001           00001   640          LA 2,1
A2F6           00544   641          ST 2,ENTFLG          SETTING THE ENTRY SEQUENCE FLAG NEGATIVE
1000           00000   642 PLOT2    LM 2,4,0(1)          ENTFLG IS THE ENTRY SEQUENCE FLAG
2000           00000   643          LE    0,0(0,2)       LOAD LINKAGE                                PLOT1240
3000           00000   644          LE    2,0(0,3)       PICK UP X                                   PLOT1250
4000           00000   645          L     4,0(0,4)       PICK UP Y                                   PLOT1260
A35E           005AC   646          LM    1,3,BUFID      PICK UP IC IN R4                            PLOT1270
A32A           00578   647          ST    4,NEWIC        LOAD BUF INFORMATION                        PLOT1280
                       648          LPR   4,4            NEWIC=IC                                    PLOT1290
A38C           005DA   649          CH    4,TEN          IC=ABS(IC)                                  PLOT1300
A0A4           002F2   650          BC    4,TESTIC       IS IC LSS 10                                PLOT1310
                       651 *   CODE  FOR  OFFSET ENTRY      YES,BRANCH TO TEST IC                   PLOT1320
A38E           005DC   652          CH    4,THTY    *                                                PLOT1330
A07E           002CC   653          BC    10,ICGT20 *    IS IC LSS 20                                PLOT1340
A38C           005DA   654          SH    4,TEN     *    NO,BRANCH TO IC GT 20                       PLOT1350
A06E           002BC   655          SE    0,XOFF    *    YES,IC=IC-10                                PLOT1360
A072           002C0   656          DE    0,XFCT    *     X=(X-XOFF)/XFCT                            PLOT1370
A076           002C4   657          SE    2,YOFF    *                                                PLOT1380
A07A           002C8   658          DE    2,YFCT    *     Y=(Y-YOFF)/YFCT                            PLOT1390
A0A4           002F2   659          BC    15,TESTIC *                                                PLOT1400
0000                   660 XOFF     DC    E'0.0'    *     GO TO TEST IC                              PLOT1410
0000                   661 XFCT     DC    E'1.0'    *    OPTIONAL FOR OFFSET ENTRY                   PLOT1420
0000                   662 YOFF     DC    E'0.0'    *                                                PLOT1430
0000                   663 YFCT     DC    E'1.0'    *                                                PLOT1440
                       664 **                                                                        PLOT1450
A390           005DE   665 ICGT20   CH    4,THTY         IS IC LSS 30                                PLOT1460
A092           002E0   666          BC    10,ICGT30        NO,BRANCH TO IC GT 30                     PLOT1470
A32A    00578          667          OI    NEWIC,X'80'      YES,SET SIGN BIT ON,BUT LEAVE NEWIC       PLOT1480
A38E           005DC   668          SH    4,THTY             IC=IC-20                                PLOT1490
A0A4           002F2   669          BC    15,TESTIC          GO TO TEST IC                           PLOT1500
A394           005E2   670 ICGT30   STH   4,BLKN         BLKN=IC                                     PLOT1510
                       671          LNR   4,4                                                       PLOT1520
A32A           00578   672          ST    4,NEWIC        NEWIC=-IC                                   PLOT1530
0003           00003   673          LA    4,3            IC=3                                        PLOT1540
A271    0043F          674          MVI   CLSWT+1,X'F0'  SET CLOSE SWITCH ON                         PLOT1550
A336           00584   675 TESTIC   C     4,ICOLD            TEST IC                                 PLOT1560
A0F8           00336   676          BC    8,TESTX        IF IC=ICOLD,BRANCH TO TEST X                PLOT1570
A336           00584   677          ST    4,ICOLD        ICOLD=IC                                    PLOT1580
A38A           005D8   678          CH    4,TWO                                                     PLOT1590
A0E8           00336   679          BC    4,TESTX        IF IC LSS 2,BRANCH TO TEST X                PLOT1600
A0C8           00316   680          BC    2,MVPNUP       IF IC GTR 2,BRANCH TO MOVE PEN UP           PLOT1610
A6D5           00923   681          IC    9,PNDNCD       IF IC = 2,LOAD CD=PEN DOWN CODE             PLOT1620
A61E           0086C   682          LH    8,PNDNCT                 LOAD CNT=PEN DOWN COUNT           PLOT1630
A0D0           0031E   683          BC    15,STPEN                 GO TO STORE PEN                   PLOT1640
A6D4           00922   684 MVPNUP   IC    9,PNUPCD             LOAD CD=PEN UP CODE                   PLOT1650
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
 A620              0086E   685           LH    8,PNUPCT          LOAD CNT=PEN UP COUNT                          PLOT1660
 1000              00000   686 STPEN     STC   9,0(0,1)         STORE CD IN BUF                                PLOT1670
 A0E0              0032E   687           BXLE  1,2,TSTPCT        INCREASE BUF AND BRANCH IF NOT FULL            PLOT1680
 A4B0              006FF   688           BAL    11,WBUF          WRITE BUF IF FULL
 A0E4              00332   689           BC    15,SVPME          BRANCH TO SAVE PEN CODE                        PLOT1700
 A0D0              0031E   690 TSTPCT    BCT   8,STPEN           DECREASE COUNT AND BRANCH IF NOT 0             PLOT1710
 A396              005E4   691 SVPME     STC   9,PMOVE           SAVE CURRENT PEN CODE                          PLOT1720
                           692 TESTX     SR    4,4               DX=0                                  R4-DX    PLOT1730
                           693           SR    6,6               K=0                                   R6-K     PLOT1740
 A33A              00588   694           CE    0,XOLD            IS X=XOLD                                      PLOT1750
 A188              00406   695           BC    8,TESTY             YES,BRANCH TO TEST Y                         PLOT1760
 A33A              00588   696           STE   0,XOLD              NO,XOLD=X                                    PLOT1770
                           697           LPER  4,0                                                              PLOT1780
 A356              C05A4   698           ME    4,FACSZ           X=X*FACSZ                                      PLOT1790
 A39A              005E8   699           CE    4,CHECKX          CHECK SIZE OF X
 A12E              0037C   700           BNP   ROUNDX            SIZE OK
                           701           WTO   'X PLOTTER LIMIT EXCEEDED'
                           702+          CNOP  0,4
 A126              00374   703+          BAL   1,IHB0006A BRANCH AROUND MESSAGE
                           704+          DC    AL2(28) TEXT LENGTH
                           705+          DC    B'0000000000000000' MCS FLAGS
D7D3D6E3E3C5               706+          DC    C'X PLOTTER LIMIT EXCEEDED'
                           707+IHB0006A DS    0H
                           708+          SVC   35
                           709           ABEND 100    X LARGER THAN 100 IN.
                           710+          DS    0H
 0064              00064   711+          LA    1,100 LOAD PARAMETER REG 1
                           712+          SVC   13 LINK TO ABEND ROUTINE
 A352              005A0   713 ROUNDX    AE    4,RNDCON          ROUND AND FIX X
 A34A              00598   714           AW    4,FXCON                                                          PLOT1810
 A342              C0590   715           STD   4,TEMP                                                           PLOT1820
 A346              00594   716           L     7,TEMP+4                                                         PLOT1830
                           717           LTER  0,0                                                              PLOT1840
 A146              00394   718           BC    11,CALCDX                                                        PLOT1850
                           719           LNR   7,7                                                              PLOT1860
                           720 *
                           721 *  THIS CODE ENSURES THAT NO WATFIV PLOT BATCH PROCESSING CONVENTIONS
                           722 *  ARE VIOLATED.  ERROR FLAGS ARE SET IF ANY ARE.  THE USER MUST
                           723 *  NOT PLOT TO THE LEFT OF HIS STARTING POSITION AND MUST FINISH THE
                           724 *  PLOT WITH THE PEN SITUATED FURTHER RIGHT THAN THE RIGHTMOST POINT
                           725 *  PLOTTED IN THE PLOT.
                           726 *
                           727 *    BASE - LOCAL X ORIGIN POINT
                           728 *    GBLPT - ACTUAL PLOT POSITION MEASURED IN PLOT STEPS (IE. 100/INCH)
                           729 *    LCLPT - PLOT POSITION RELATIVE TO CURRENT ORIGIN.
                           730 *    REGISTER USAGE
                           731 *      9 - CONTAINS GBLPT.
                           732 *      8 - WORK REGISTER USED IN SETTING ERROR FLAGS.
 A302              00550   733 CALCDX    L 9,BASE
                           734           AR 9,7                  GBLPT = BASE + LCLPT
 A271      0048F           735           CLI CLSWT+1,X'F0'       CHECKING FOR 'END OF PLOT'
 A176              003C4   736           BNE CKNORG              NO. CHECK FOR NEW ORIGIN
                           737 *  DETERMINING FINAL Y COORDINATE POSITION
 A2FE              0054C   738           LE 2,TTWO
                           739           LNER 2,2
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
A30A            00558   740           C 9,RXP             CHECKING IF (GBLPT >= RXP)
A16E            003BC   741           BC 11,D11           YES.
0001            00001   742           LA 8,1
A312            00560   743           ST 8,WNFLG          SET WARNING MESSAGE FLAG
A30A            00558   744           L 9,RXP
                        745 *  DETERMINING COORDINATE LOCATION TO CLOSE FILE AT.
9190            00190   746 D11       LA 7,400(0,9)
A1A4            003F2   747           BC 15,NEXTDX
A32A            00578   748 CKNORG    L 8,NEWIC           CHECKING FOR A NEW ORIGIN
                        749           LTR 8,8             IS NEWIC NEGATIVE
A184            003D2   750           BC 11,TESTL         NO.
A302            00550   751           ST 9,BASE           BASE = BASE + LCLPT
A306            00554   752 TESTL     C 9,LXP             IS GBLPT < LXP ?
A198            003E6   753           BC 11,TESTUP        NO.
0001            00001   754           LA 8,1
A30F            0055C   755           ST 8,ERFLG          SETTING ERROR FLAG.
A274            004C2   756           BC 15,EXIT          RETURN WITHOUT DOING PLOT
A30A            00558   757 TESTUP    C 9,RXP             IS GBLPT > RXP ?
A1A4            003F2   758           BC 13,NEXTDX        NO.
A30A            00558   759           ST 9,RXP            RXP = GBLPT
                        760 NEXTDX    LR 4,7
A32E            0057C   761           S 4,NPX             DX=X-NPX
A32E            0057C   762           ST    7,NPX          NPX=X                                    PLOT1890
A188            00406   763           BC    10,TESTY       IF DX IS POSITIVE,BRANCH TO TEST Y       PLOT1900
6001            00001   764           LA    6,1(0,6)       K=K+1                                    PLOT1910
                        765           LPR   4,4            DX=-DX                                   PLOT1920
                        766 TESTY     SR    5,5            DY=0                              R5-DY  PLOT1930
A33E            0058C   767           CE    2,YOLD         IS Y=YOLD                                PLOT1940
A22A            00478   768           BC    8,TESTDX         YES,BRANCH TO TEST DXDY                PLOT1950
A33E            0058C   769           STE   2,YOLD           NO,YOLD=Y                              PLOT1960
                        770           LPER  4,2                                                     PLOT1970
A356            005A4   771           ME    4,FACSZ        Y=Y*FACSZ                              PLOT1980
A39E            005EC   772           CE    4,CHECKY       CHECK SIZE OF Y
A1FE            0044C   773           BNP   ROUNDY         SIZE OK
                        774           WTO   'Y PLOTTER LIMIT EXCEEDED'
                        775+          CNOP  0,4
A1F6            00444   776+          BAL   1,IH80009A BRANCH AROUND MESSAGE
                        777+          DC    AL2(28) TEXT LENGTH
                        778+          DC    8'0000000000000000' MCS FLAGS
7D3D6E3E3C5             779+          DC    C'Y PLOTTER LIMIT EXCEEDED'
                        780+IH80009A  DS    0H
                        781+          SVC   35
                        782           ABEND 30    Y LARGER THAN 30 IN.
                        783+          DS    0H
001E            001E    784+          LA    1,30 LOAD PARAMETER REG 1
                        785+          SVC   13 LINK TO ABEND ROUTINE
A352            005A0   786 ROUNDY    AE    4,RNDCON       ROUND AND FIX Y
A34A            00598   787           AW    4,FXCON                                                PLOT2000
A342            00590   788           STD   4,TEMP                                                 PLOT2010
A346            00594   789           L     7,TEMP+4                                               PLOT2020
                        790           LTER  2,2                                                    PLOT2030
A216            00464   791           BC    11,CALCDY                                              PLOT2040
                        792           LNR   7,7                                                    PLOT2050
                        793 CALCDY    LR    5,7                                                    PLOT2060
A332            00580   794           S     5,NPY          DY=Y-NPY                                PLOT2070
```

```
T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    PLOT2080
A332            00580   795            ST    7,NPY          NPY=Y                                 PLOT2090
A22A            00478   796            BC    10,TESTDX      IF DY IS POSITIVE,BRANCH TO TEST DXDY PLOT2100
6002            00002   797            LA    6,2(0,6)       K=K+2                                 PLOT2110
                        798            LPR   5,5            DY=-DY                                PLOT2120
                        799 TESTDX     LR    7,4                                                  PLOT2130
                        800            OR    7,5            IF DX AND DY ARE BOTH ZERO            PLOT2140
A246            00494   801            BC    8,ENDIC              BRANCH TO ENDIC                 PLOT2150
                        802            CR    4,5                                                  PLOT2160
A3A6            005F4   803            BC    11,PLTALG      IF DX GEQ DY,BRANCH TO PLOT ALG       PLOT2170
                        804            LR    7,4                                                  PLOT2180
                        805            LR    4,5            EXCHANGE DX AND DY                    PLOT2190
                        806            LR    5,7                                                  PLOT2200
6004            00004   807            LA    6,4(0,6)       K=K+4                                 PLOT2210
A3A6            005F4   808            BC    15,PLTALG      BRANCH TO PLOT ALG                    PLOT2220
                        809 ENDIC      SR    0,0            RETURN FROM PLOT ALG                  PLOT2230
A32A    00578           810            TM    NEWIC,X'FF'    IF SIGN BIT OF NEWIC IS OFF           PLOT2240
A274            004C2   811            BC    8,EXIT         BRANCH TO EXIT                        PLOT2250
A32E            0057C   812            ST    0,NPX          NPX=0                                 PLOT2260
A332            00580   813            ST    0,NPY          NPY=0                                 PLOT2270
A336            00584   814            ST    0,ICOLD        ICOLD=0                               PLOT2280
A33A            00588   815            ST    0,XOLD         XOLD=0                                PLOT2290
A33E            0058C   816            ST    0,YOLD         YOLD=0                                PLOT2300
A274            004C2   817            BC    4,EXIT         IF NEWIC IS POSITIVE BRANCH TO EXIT
A4B0            006FE   818            BAL   11,WBUF        WRITE PARTIAL BUFFER
A3D6            00624   819            BAL   11,WBLK        WRITE BLOCK ADDRESS
A544            00792   820 CLSWT      BC    0,CLTAP        IF CLOSE SWITCH IS ON,BRANCH TO CLOSE PLOT2330
                                                                                                   PLOT2340
A35E    -       005AC   822 EXIT       STM   1,3,BUFID      SAVE BUF INFORMATION
                        823 *
                        824 *    DECIDING WHICH RETURN SEQUENCE TO USE
                        825 *    A WATFIV RETURN, OR AN OS-TYPE RETURN
                        826 *          A WATFIV RETURN OR AN OS-TYPE RETURN
                        827 *
A2F6            00544   828            L 3,ENTFLG           TESTING THE ENTRY SEQUENCE FLAG.
                        829            LTR 3,3              IF FLAG IS ZERO
A292            C04E0   830            BC 8,WATRET          BRANCH TO WATFIV RETURN SEQUENCE
                        831            SR 3,3
A2F6            00544   832            ST 3,ENTFLG          RESETTING FLAG TO ZERO
A29A            004E8   833            L 13,SAVE+4
D00C    -       0000C   834            LM 14,12,12(13)
                        835            BR 14                USING OS-TYPE RETURN
                        836 *   A  WATFIV  TYPE  RETURN
C464            00464   837 WATRET     BC 15,XRET(12)       BRANCH TO XRET RTN.
                        838            DS OF
                        839 SAVE       DS    24F          * CONSTANTS AND VARIABLES
0000                    840 ENTFLG     DC F'0'
C464            00464   841 AGBZ       BC 15,1124(0,12)
0000                    842 TTWO       DC E'11.'
0000                    843 BASE       DC F'0'
0000                    844 LXP        DC F'0'
0000                    845 RXP        DC F'0'
0000                    846 ERFLG      DC F'0'
0000                    847 WNFLG      DC F'0'
0000                    848 ADERR      DC A(ERROR1)
0000                    849 EMN        DC F'0'
```

```
F CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                                    PLOT2400
                        850 SVR3     DS    3F                                                                      PLOT2410
000                     851 NEWIC    DC     F'0'                                                                   PLOT2420
000                     852 NPX      DC     F'0'                                                                   PLOT2430
000                     853 NPY      DC     F'0'                                                                   PLOT2440
000                     854 ICOLD    DC     F'0'                                                                   PLOT2450
000                     855 XOLD     DC     F'0'                                                                   PLOT2460
000                     856 YOLD     DC     F'0'                                                                   PLOT2470
00000000000             857 TEMP     DC     D'0'                                                                   PLOT2480
00000000000             858 FXCON    DC     X'4E000000000000000'                                                   PLOT2490
000                     859 RNDCON   DC     E'0.5'                                                                 PLOT2500
000                     860 FACSZ    DC     E'0.0'                                                                 PLOT2510
000                     861 FACT     DC     E'1.0'                                                                 PLOT2520
00100000001             862 BUFID    DC     3F'1'      1-NEXT LOC IN BUFFER,1-ONE,3-LAST LOC IN BUFFER             PLOT2530
00100000001             863 BUFA     DC     4F'1'       ININTIAL BUFID AND BUFLK VALVES FOR 1ST BUFFER              PLOT2540
001                     864 BUFLK    DC     F'1'        STARTING LOCATION OF NEXT RECORD                            PLOT2550
001                     865 BUFFL    DC     F'1'        FIRST DATA LOCATION IN NEXT RECORD                          PLOT2560
001                     866 BUFCNT   DC     F'1'        COUNT FOR LAST RECORD WRITTEN                               PLOT2570
FFF                     867 ADDMSK   DC     X'00FFFFFF'                                                            PLOT2580
                        868 TWO      DC     H'2'                                                                   PLOT2590
                        869 TEN      DC     H'10'                                                                  PLOT2600
                        870 TWTY     DC     H'20'                                                                  PLOT2610
                        871 THTY     DC     H'30'                                                                  PLOT2620
                        872 LGLDEV   DC     H'11'                                                                  PLOT2630
                        873 BLKN     DC     H'1'                                                                   PLOT2640
                        874 PMOVE    DC     H'0'
                        875 N999     DC     H'999'
000                     876 CHECKX   DC     E'10000.'     MAX SIZE OF X*100
000                     877 CHECKY   DC     E'3000.'      MAX SIZE OF Y*100
A00                     878 XMAX     DC     E'100000.'       XMAX=1000 IN.
                        879 **                                                                                     PLOT2650
                        880 *    CODE  FOR  8 VECTOR ROUTINES WITHOUT NOP                                          PLOT2660
                        881 PLTALG   LR     9,4            NC=DX                          R4=NC                    PLOT2670
                        882          LR     8,5            NR=2*DX                        R5=CBCD                  PLOT2680
                        883          LR     7,4            NA=DX                          R6=MJCD                  PLOT2690
0001            00001   884          SLDA   8,1            NT=2*DX                        R7=NA                    PLOT2700
A6DE            0092C   885          IC     5,CMBCD(6)     CBCD=CMBCD(J)                  R8=NR                    PLOT2710
A6D6            00924   886          IC     6,MAJCD(6)     MJCD=MAJCD(J)                  R9=NT                    PLOT2720
1000            00000   887 STLOOP   STC    6,0(0,1)       STORE MJCD IN BUF                                       PLOT2730
A3C6            00614   888          BXLE   7,8,STBF       NA=NA+NR, IF NA LSS NT,BRANCH TO STBF                   PLOT2740
                        889          SR     7,9            NA=NA-NT                                                PLOT2750
1000            00000   890          STC    5,0(0,1)       STORE CBCD IN BUF                                       PLOT2760
A3CE            0061C   891 STBF     BXLE   1,2,*+8        TEST FOR FULL BUFFER                                    PLOT2770
A4B0            006FE   892          BAL    11,WBUF         WRITE
A3B8            00606   893          BCT    4,STLOOP       NC=NC-1,IF NC GTR 0,BRANCH TO REPEAT                    PLOT2790
A246            00494   894          BC     15,ENDIC         EXIT                                                  PLOT2800
                        895 **                                                                                     PLOT2810
A35C            005AC   896 WBLK     STM    1,3,BUFID       SAVE BUF INFORMATION                                   PLOT2820
A394            005E2   897          LH     3,BLKN                                                                 PLOT2830
A342            00590   898          CVD    3,TEMP          CONVERT BLKN TO DECIMAL                                PLOT2840
3001            00001   899          LA     3,1(0,3)                                                               PLOT2850
A394            005E2   900          STH    3,BLKN          INCREASE BLKN BY ONE                                   PLOT2860
A346            00594   901          L      2,TEMP+4        PICK UP DECIMAL BLKN                                   PLOT2870
0004            00004   902          SRL    2,4             SHIFT OFF SIGN                                         PLOT2880
0004            00004   903          SRDL   2,4             SHIFT UNITS DIGIT AND                                  PLOT2890
0018            00018   904          SRL    3,27            DOUBLE                                                 PLOT2900
```

```
:T CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    PLOT2910
 A626             00874   905           LH    5,BKTAB(3)      PICK UP CODES FOR UNITS DIGIT       PLOT2920
 A679             0C8C7   906           STC   5,BKNO+15                                           PLOT2930
 0008             00008   907           SRL   5,8             AND STORE CODES IN BLOCK RECORD      PLOT2940
 A676             008C4   908           STC   5,BKNO+12                                          PLOT2950
 0004             00004   909           SRDL  2,4             SHIFT TENS DIGIT AND                PLOT2960
 0018             00018   910           SRL   3,27            DOUBLE                              PLOT2970
 A626             00874   911           LH    5,BKTAB(3)      PICK UP CODES FOR TENS DIGIT        PLOT2980
 A673             0C8C1   912           STC   5,BKNO+9                                           PLOT2990
 0008             00008   913           SRL   5,8             AND STORE CODES IN BLOCK RECORD      PLOT3000
 A670             0088E   914           STC   5,BKNO+6                                           PLOT3010
 0004             00004   915           SRDL  2,4             SHIFT HUNDREDS DIGIT AND            PLOT3020
 0018             00018   916           SRL   3,27            DOUBLE                              PLOT3030
 A626             00874   917           LH    5,BKTAB(3)      PICK UP CODES FOR HUNDREDS DIGIT    PLOT3040
 A66D             008B8   918           STC   5,BKNO+3                                           PLOT3050
 0008             00008   919           SRL   5,8             AND STORE CODES IN BLOCK RECORD      PLOT3060
 A66A             008B8   920           STC   5,BKNO                                             PLOT3070
 A63A             00888   921           LA    4,BKCD          LOAD ADDR AND COUNT FOR WRITE       PLOT3080
 A622             00870   922           LH    5,BKCNT                                            PLOT3090
                          923           WRITE  BLKCB,SF,PTDCB,(4),(5)
                          924+          CNOP  0,4
 A45A             006A8   925+          BAL   1,*+24 LOAD DECB ADDRESS
 0000                     926+BLKCB     DC    F'0' EVENT CONTROL BLOCK
                          927+          DC    X'00' TYPE FIELD
                          928+          DC    X'20' TYPE FIELD
                          929+          DC    AL2(0) LENGTH
 098C                     930+          DC    A(PTDCB) DCB ADDRESS
 0000                     931+          DC    A(0) AREA ADDRESS
 0000                     932+          DC    A(0) RECORD POINTER WORD
 000C             0000C   933+          ST    4,12(1,0) STORE AREA ADDRESS
 0006             00006   934+          STH   5,6(1,0) STORE LENGTH
 0008             00008   935+          L     15,8(1,0) LOAD DCB ADDRESS
 F030             00030   936+          L     15,48(0,15) LOAD RDWR ROUTINE ADDR
                          937+          BALR  14,15 LINK TO RDWR ROUTINE
                          938           CHECK  BLKCB                                             PLOT3100
 A446             0C694   939+          LA    1,BLKCB LOAD PARAMETER REG 1
 1008             00008   940+          L     14,8(0,1) PICK UP DCB ADDRESS
 E034             00034   941+          L     15,52(0,14) LOAD CHECK ROUT. ADDR.
                          942+          BALR  14,15 LINK TO CHECK ROUTINE
                          943 WDUMCB    WRITE DUMCB,SF,PTDCB,SYCD,10                             PLOT3110
                          944+          CNOP  0,4
 A492             0C6E0   945+WDUMCB    BAL   1,*+24 LOAD DECB ADDRESS
 0000                     946+DUMCB     DC    F'0' EVENT CONTROL BLOCK
                          947+          DC    X'00' TYPE FIELD
                          948+          DC    X'20' TYPE FIELD
                          949+          DC    AL2(10) LENGTH
 098C                     950+          DC    A(PTDCB) DCB ADDRESS
 08E2                     951+          DC    A(SYCD) AREA ADDRESS
 0000                     952+          DC    A(0) RECORD POINTER WORD
 0008             00008   953+          L     15,8(1,0) LOAD DCB ADDRESS
 F030             00030   954+          L     15,48(0,15) LOAD RDWR ROUTINE ADDR
                          955+          BALR  14,15 LINK TO RDWR ROUTINE
                          956           CHECK DUMCB                                              PLOT3120
 A47E             006CC   957+          LA    1,DUMCB LOAD PARAMETER REG 1
 1008             00008   958+          L     14,8(0,1) PICK UP DCB ADDRESS
 E034             00034   959+          L     15,52(0,14) LOAD CHECK ROUT. ADDR.
```

```
T CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                        960+          BALR  14,15 LINK TO CHECK ROUTINE
A35E               005AC 961           LM    1,3,BUFID      RESTORE BUF INFORMATION                  PLOT3130
                        962           BCR   15,11    EXIT
                        963 **                                                                        PLOT3150
A36A               005B8 964 WBUF      C     1,BUFA         IF POINTER IS AT FIRST LOC IN BUFFER     PLOT3160
                        965           BCR   8,11      THEN EXIT
A31E               0056C 966           STM   4,6,SVR3       SAVE R4-6                                PLOT3180
                        967           LR    3,1            SET POINTER TO                           PLOT3190
                        968           BCTR  3,0              LAST LOC FILLED IN BUFFER              PLOT3200
A400               0071E 969 BUFCK     BC    15,BUFNC       BRANCH IF CHECK FLAG OFF                 PLOT3210
                        970           CHECK BUFCB                                                     PLOT3220
A512               00760 971+          LA    1,BUFCB LOAD PARAMETER REG 1
1008               00008 972+          L     14,8(0,1) PICK UP DCB ADDRESS
F034               00034 973+          L     15,52(0,14) LOAD CHECK ROUT. ADDR.
                         974+          BALR  14,15 LINK TO CHECK ROUTINE
A48F       0070D         975 BUFNC     MVI   BUFCK+1,0      SET CHECK FLAG ON                        PLOT3230
A36A               00588 976           L     1,BUFA         SET TO FIRST LOCATION IN BUFFER          PLOT3240
A37E               005CC 977           L     4,BUFFL        SET TO FIRST DATA LOC IN OUTPUT          PLOT3250
                        978           SR    6,6                                                       PLOT3260
1000               00000 979 WSTR      IC    6,0(0,1)                                                PLOT3270
A6E6               00934 980           LA    5,CDTAB(6)                                              PLOT3280
4000 5000 00000    00000 981           MVC   0(NOB,4),0(5)                                           PLOT3290
4009               00009 982           LA    4,NOB(0,4)                                              PLOT3300
A4DE               0072C 983           BXLE  1,2,WSTR                                                PLOT3310
4000 A6CA 00000    00713 984           MVC   0(NOE,4),ENDPL   MOVE END PLOT CODES INTO BUFFER        PLOT3320
400A               0000A 985           LA    4,NOE(0,4)                                              PLOT3330
A37A               005C8 986           S     4,BUFLK        CALCULATE LENGTH OF RECORD               PLOT3340
A382               005D0 987           ST    4,BUFCNT        AND STORE                               PLOT3350
A37A               005C8 988           L     4,BUFLK        LOAD ADDRESS AND COUNT FOR WRITE         PLOT3360
A382               005D0 989           L     5,BUFCNT                                                PLOT3370
                        990           WRITE BUFCB,SF,PTDCB,(4),(5)                                    PLOT3380
                        991+          CNOP  0,4
A526               00774 992+          BAL   1,*+24 LOAD DECB ADDRESS
0000                     993+BUFCB     DC    F'0' EVENT CONTROL BLOCK
                         994+          DC    X'00' TYPE FIELD
                         995+          DC    X'20' TYPE FIELD
                         996+          DC    AL2(0) LENGTH
098C                     997+          DC    A(PTDCB) DCB ADDRESS
0000                     998+          DC    A(0) AREA ADDRESS
0000                     999+          DC    A(0) RECORD POINTER WORD
000C               0000C 1000+         ST    4,12(1,0) STORE AREA ADDRESS
0006               00006 1001+         STH   5,6(1,0) STORE LENGTH
0008               00008 1002+         L     15,8(1,0) LOAD DCB ADDRESS
F030               00030 1003+         L     15,48(0,15) LOAD RDWR ROUTINE ADDR
                         1004+         BALR  14,15 LINK TO RDWR ROUTINE
A36A               00588 1005          LM    1,3,BUFA       LOAD BUFFER CONSTANTS                    PLOT3390
A31E               0056C 1006          LM    4,6,SVR3       RESTORE R4-6                             PLOT3400
                         1007          BCR   15,11    EXIT
                         1008          CNOP  2,4                                                       PLOT3420
                         1009 **
                         1010 *  THIS CODE GENERATES ERROR MESSAGES OCCURING WHEN PROPER WATFIV
                         1011 *  BATCH PROCESSING PLOTTING CONVENTIONS ARE NOT FOLLOWED.  THE USER
                         1012 *  MUST POSITION THE PEN TO THE RIGHT OF HIS PLOT AND MUST HAVE MADE
                         1013 *  NO ATTEMPTS TO PLOT TO THE LEFT OF THE INITIAL STARTING POSITION.
0001               00001 1014 CLTAP    LA 4,1
```

```
T CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
A30E             0055C  1015           C 4,ERFLG           IS ERROR FLAG ON?
A56E             0078C  1016           BC 7,WARN+4         NO
0002             00002  1017           LA 5,2              YES
A31A             00568  1018           ST 5,EMN            STORING ERROR MESSAGE NUMBER
                        1019 *  SETTING UP REGISTERS FOR THE CALL TO ERROR1
A56A             007B8  1020           LA 14,WARN          LOADING RETURN ADDRESS
A316             00564  1021 CALL1     L 15,ADERR          ADDRESS OF ERROR ROUTINE
                        1022           BALR 1,15
                        1023 *  THE CALLING ARGUMENT LIST
568                     1024           DC X'82',AL3(EMN)
000                     1025           DC X'10',AL3(0)
                        1026 *  CHECKING FOR A WARNING MESSAGE
D00C             0000C  1027 WARN      LM 14,11,12(13)
0001             00001  1028           LA 4,1
A312             00560  1029           C 4,WNFLG           CHECKING FOR A WARNING MESSAGE
A58A             0C7D8  1030           BC 7,FINI+4         NO WARNING MESSAGE
0003             00003  1031           LA 5,3
A31A             00568  1032           ST 5,EMN            STORING WARNING MSG NUMBER
                        1033 *  SETTING UP REGISTERS FOR THE CALL TO ERROR1
A55C             007AA  1034           BAL 14,CALL1        BRANCH TO WATFIV CALLING INSTRUCTION
D00C             0000C  1035 FINI      LM 14,11,12(13)
                        1036           CLOSE (PTDCB)
                        1037+          CNOP  0,4
A592             0C7E0  1038+          BAL   1,*+8 BRANCH AROUND LIST
                        1039+          DC    AL1(128) OPTION BYTE
C                       1040+          DC    AL3(PTDCB) DCB ADDRESS
                        1041+          SVC   20 ISSUE CLOSE SVC
0010             00010  1042           L 1,16
0000             00000  1043           L     1,0(1)                TCBP ADDR
0004             00004  1044           L     1,4(1)                TCB ADDR
000C             0000C  1045           L     1,12(1)               TIOT ADDR
A764 1000 00982  00000  1046           MVC   JOBNAME(8),0(1)       JOB NAME
A74A             00998  1047           LA    1,WTOMSG              -> MSG
                        1048           SVC   35                    ISSUE WTO
A274             004C2  1049           BC    15,EXIT                                    PLOT3440
                        1050 **                                                         PLOT3450
                        1051 OPTAP     LPR   3,3                                        PLOT3460
0002             00002  1052           SLA   3,2           CHANGE COUNT TO BYTES        PLOT3470
A524             00872  1053           CH    3,BLKMAX      COMPARE AGAINST MAXIMUM      PLOT3480
A5C6             0C914  1054           BC    12,*+8                                     PLOT3490
A624             00872  1055           LH    3,BLKMAX      IF TOO LARGE REPLACE WITH MAXIMUM PLOT3500
A7AC             009FA  1056           STH   3,PTDCB+62    SET LRECL PARAMETER IN DCB   PLOT3510
2000             00000  1057           LA    7,0(3,2)      SET TO LAST LOC IN BUFFER    PLOT3520
0036             00036  1058           LA    4,NOS(0,0)    CALCULATE                    PLOT3530
400A             0000A  1059           LA    4,NOE(0,4)     THE NUMBER                  PLOT3540
                        1060           SR    3,4             OF BYTES                   PLOT3550
                        1061           LR    5,3               IN THE                   PLOT3560
                        1062           SR    4,4                FIRST BUFFER            PLOT3570
000A             0C00A  1063           LA    3,NOB+1(0,0)        AS FOLLOWS             PLOT3580
                        1064           DR    4,3           K=(COUNT-NOE-NOS)/(NOB+1)    PLOT3590
A36A             00588  1065           ST    2,BUFA        BUFA(1)=BUF                  PLOT3600
                        1066           AR    2,5                                        PLOT3610
A37A             005C8  1067           ST    2,BUFLK       BUFLK=BUF+K                  PLOT3620
2000 A694 00000  008E2  1068           MVC   0(NOS,2),SYCD MOVE SYNC CODES INTO BUF+K   PLOT3630
                        1069           BCTR  2,0                                        PLOT3640
```

```
CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
372            005C0  1070           ST    2,BUFA+8        BUFA(3)=BUF+K-1                  PLOT3650
037            00037  1071           LA    2,NOS+1(0,2)                                     PLOT3660
37F            005CC  1072           ST    2,BUFFL         BUFFL=BUF+K+NOS                  PLOT3670
                      1073 **                                                               PLOT3680
                      1074           OPEN (PTDCB,OUTPUT)
                      1075+          CNOP  0,4
60A            00858  1076+          BAL   1,*+8 LOAD REG1 W/LIST ADDR.
                      1077+          DC    AL1(143) OPTION BYTE
                      1078+          DC    AL3(PTDCB) DCB ADDRESS
                      1079+          SVC   19 ISSUE OPEN SVC
3D6            00624  1080           BAL   11,WBLK         WRITE BLOCK ADDRESS 1
36A            00588  1081           LM 1,3,BUFA         LOAD BUF INFORMATION
                      1082 **                                                               PLOT3720
274            004C2  1083           BC    15,EXIT         EXIT                             PLOT3730
                      1084 **                                                               PLOT3740
                      1085 **                                                               PLOT3750
                      1086           DS    0F                                               PLOT3760
56                    1087 ADPLT     DC A(PLOT1)
                      1088 PNDNCT    DC    H'18'           NO. OF PEN DOWN CODES            PLOT3770
                      1089 PNUPCT    DC    H'03'           NO. OF PEN UP CODES              PLOT3780
                      1090 BKCNT     DC    H'114'          LENGTH OF BLOCK ADDRESS RECORD   PLOT3790
                      1091 BLKMAX    DC    H'32767'        MAXIMUM RECORD LENGTH            PLOT3800
                      1092 NOS       EQU   54              NO. OF SYNC. CODES               PLOT3810
                      1093 NOE       EQU   10              NO. OF END PLOT CODES            PLOT3820
                      1094 NOB       EQU   9               BUFFER SPLIT FACTOR              PLOT3830
81                    1095 BKTAB     DC    X'90909081'     BLOCK ADDRESS CODES- 0  1        PLOT3840
F3                    1096           DC    X'90D290F3'                          2  3        PLOT3850
81                    1097           DC    X'819081B1'                          4  5        PLOT3860
F3                    1098           DC    X'81D281F3'                          6  7        PLOT3870
81                    1099           DC    X'D290D2B1'                          8  9        PLOT3880
9000009000            1100 BKCD      DC   8X'900000'       BLOCK ADDRESS RECORD             PLOT3890
6300006300            1101           DC   7X'630000'                                        PLOT3900
                      1102           DC    X'210000'                                        PLOT3910
9000009000            1103 BKND      DC   6X'900000'                                        PLOT3920
                      1104           DC    X'210000'                                        PLOT3930
6300006300            1105           DC   7X'630000'                                        PLOT3940
9000009000            1106 SYCD      DC  10X'900000'       SYNC CODES FOR DATA RECORD       PLOT3950
6300006300            1107           DC   7X'630000'                                        PLOT3960
                      1108           DC    X'420000'                                        PLOT3970
0200                  1109 ENDPL     DC    X'900000D200'   END PLOT CODES                  PLOT3980
0090                  1110           DC    X'00630000090'                                   PLOT3990
                      1111 PNUPCD    DC    X'00'           PEN UP CODE                      PLOT4000
                      1112 PNDNCD    DC    X'09'           PEN DOWN CODE                    PLOT4010
                      1113 * CODE TABLES FOR VALUES OF J=     *FOR J=0 THE                   PLOT4020
                      1114 *                0 1 2 3 4 5 6 7  *CODES ARE                     PLOT4030
4812123636            1115 MAJCD     DC    X'2448244812123636'   +X                       PLOT4040
3F18512D3F            1116 CMBCD     DC    X'18512D3F18512D3F'   +X+Y                     PLOT4050
D200008100            1117 CDTAB     DC    X'D20000D20000B10000'  COMMAND FOR-PEN UP       PLOT4060
D200C0F3C0            1118           DC    X'D20000D20000F30000'              PEN DOWN      PLOT4070
F300000200            1119           DC    X'D20000F30000D20000'              +Y            PLOT4080
F300000200            1120           DC    X'F30000F30000D20000'              +Y+X          PLOT4090
D200000200            1121           DC    X'F30000D20000D20000'                +X          PLOT4100
B100000200            1122           DC    X'F30000B10000D20000'              -Y+X          PLOT4110
B100000200            1123           DC    X'D20000B10000D20000'              -Y            PLOT4120
B100000200            1124           DC    X'B10000B10000D20000'              -Y-X          PLOT4130
```

```
 CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
)D200000200            1125           DC    X'810000D20000D20000'              -X          PLOT4140
)F300000200            1126           DC    X'810000F30000D20000'             +Y-X         PLOT4150
)D200000200            1127           DC    X'D20000D20000D20000'             NOP          PLOT4160

)00D7D3D6E3            1128 WTOMSG    DC    OF'0',H'34,0',CL30'PLOT COMPLETE FOR JOB'
                       1129 JOBNAME   EQU   WTOMSG+26
                       1130 PTDCB     DCB   DSORG=PS,MACRF=W,DDNAME=PLOTTAPE,RECFM=U,NCP=3,BUFNO=1,DXPLOT4170
                                            EVD=TA                                          PLOT4180


                       1132+*                      DATA CONTROL BLOCK
                       1133+*

                       1134+PTDCB     DC    OF'0' ORIGIN ON WORD BOUNDARY

                       1136+*                      MAGNETIC TAPE DEVICE INTERFACE

)0000000000            1138+          DC    BL16'0' NERRS,NOISE,UERRS,BLKCT
                       1139+          DC    BL1'00000000' TRTCH
)                      1140+          DC    BL3'000000000000000000000000' DEVT,DEN

                       1142+*                      COMMON ACCESS METHOD INTERFACE

                       1144+          DC    AL1(1) BUFNO
                       1145+          DC    AL3(1) BUFCB
                       1146+          DC    AL2(0) BUFL
                       1147+          DC    BL2'0100000000000000' DSORG
)01                    1148+          DC    A(1) IOBAD

                       1150+*                      FOUNDATION EXTENSION

                       1152+          DC    BL1'00000000' BFTEK,BFLN,HIARCHY
                       1153+          DC    AL3(1) EODAD
                       1154+          DC    BL1'11000000' RECFM
)                      1155+          DC    AL3(0) EXLST

                       1157+*                      FOUNDATION BLOCK

)E3E3C1D7C5            1159+          DC    CL8'PLOTTAPE' DDNAME
                       1160+          DC    BL1'00000010' OFLGS
                       1161+          DC    BL1'00000000' IFLG
                       1162+          DC    BL2'0000000000100000' MACR

                       1164+*                      BSAM-BPAM-QSAM INTERFACE

                       1166+          DC    BL1'00000000' RER1
                       1167+          DC    AL3(1) CHECK, GERR, PERR
)01                    1168+          DC    A(1) SYNAD
                       1169+          DC    H'0' CIND1, CIND2
                       1170+          DC    AL2(0) BLKSIZE
)00                    1171+          DC    F'0' WCPO, WCPL, OFFSR, OFFSW
)01                    1172+          DC    A(1) IOBA
                       1173+          DC    AL1(3) NCP
                       1174+          DC    AL3(1) EOBR, EOBAD
```

```
CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                     1176+*                          BSAM-BPAM INTERFACE
01                   1178+          DC    A(1) EOBW
                     1179+          DC    H'0' DIRCT
                     1180+          DC    AL2(0) LRECL
01                   1181+          DC    A(1) CNTRL, NOTE, POINT
00                   1182 STPSZ     DC    E'100.'        NO. OF PLOTTER STEPS PER INCH     PLOT4190
                     1183           END                                                    PLOT4200
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                           1 SYMBOL   START 0         CALL SYMBOL(X,Y,HGT,IBCD,TH,N)      V127       SYMB0000
                           2 *
                           3 *   THIS SUBPROGRAM IS THE WATFIV IMPLEMENTATION VERSION OF THE CALCOMP
                           4 *   PLOT ROUTINE CALLED SYMBOL.  IT IS CALLED TO DRAW TEXT SUCH AS
                           5 *   TITLES AND TO DRAW SPECIAL CENTERED SYMBOLS.
                           6 *
                           7          EXTRN SIN1      SIN1 AND COS1 ARE THE SIN AND COS FUNCTIONS.
                           8          EXTRN COS1
                           9          EXTRN PLOT1     OS-TYPE ENTRY POINT TO PLOT ROUTINE.
                          10          ENTRY SYMB1     OS-TYPE ENTRY POINT TO SYMBOL ROUTINE
                          11 *
                          12 *
                          13 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                          14 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                          15 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                          16 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                          17 *
                          18 XENTSPEC EQU 246
                          19 XRET     EQU 1124
                          20 XA1      EQU 1592
                          21 *
                          22 *
                          23          BALR 9,0
                          24          USING *,9
                          25 *
                          26 *   CHECKING TYPE OF CALL MADE TO SYMBOL
                          27 *
) 1014            00014   28          L 2,20(0,1)          LOAD ADDRESS OF ICODE
; 2000            00000   29          L 2,0(0,2)           OBTAIN CONTENTS OF ICODE
                          30          LTR 2,2              IS IT A SPECIAL CALL
) 906C            0006E   31          BC 4,INTS            BRANCH IF SO
                          32 *   IT WAS A 'STANDARD' CALL.
                          33 *
                          34 *   DETERMINING WHETHER IBCD IS A SIMPLE VARIABLE, A VECTOR, OR A
                          35 *   HOLLERITH STRING.
                          36 *
) 100C      0000C         37          TM 12(1),X'09'       HOLLERITH STRING ?
) 9052            00054   38          BC 1,CHARK           YES
; 100C      0000C         39          TM 12(1),X'90'       VECTOR ?
; 9074            00076   40          BC 12,NEXTY          NO.  MUST BE SIMPLE VARIABLE
                          41 *
                          42 *   IBCD  IS  A  VECTOR
                          43 *
) 100C            0000C   44          L 2,12(0,1)  OBTAIN ADDRESS OF IBCD'S STAR ROUTINE
) 2004            00004   45          L 3,4(0,2)   OBTAIN ADDRESS OF THE CALLING VECTOR
; 200C            0000C   46          L 2,12(0,2)  OBTAIN THE VECTOR LENGTH FOR THE ACTUAL ARGUMENT
                          47 *  IT IS STORED IN A LOCATION REFERENCED BY THE STAR ROUTINE FOR THE
                          48 *  DUMMY ARGUMENT REPRESENTING IBCD.
) 93FE            00400   49          ST 2,TPST            STORING THE VECTOR LENGTH
                          50 *  STORING THE VECTOR ADDRESS IN THE OS-TYPE ARGUMENT LIST.
) 90F6            000E8   51          ST 3,DOPE
) 93EA            003EC   52          LA 2,STAR            OBTAIN ADDRESS OF THE STAR RTN. FOR ARRAY
) 90C6            000C8   53          ST 2,IBCD            PUTTING THIS ADDRESS IN THE MODEL ARGUMENT
                          54 *                               LIST.
                          55 *
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                           56 *  DETERMINING TYPE OF IBCD ARGUMENT
2 100C       0000C         57          TM 12(1),X'02'
0 904A             0004C   58          BC 1,INTV
4 90C6       000C8         59          MVI IBCD,X'94'     IS REAL*4 TYPE
0 90A6             000A8   60          BC 15,ENTSEQ
2 90C6       000C8         61 INTV     MVI IBCD,X'92'     IS INTEGER*4 TYPE
C 90A6             000A8   62          BC 15,ENTSEQ

                           64 *
                           65 *  IBCD  IS  A  HOLLERITH  STRING.
                           66 *
0 100C             0000C   67 CHARK    L 2,12(0,1)  OBTAIN ADDRESS OF THE ACTUAL DOPE VECTOR FOR IBCD
                           68 *  MOVING THE VECTOR LENGTH INTO THE DOPE VECTOR LENGTH FIELD
                           69 *  OF THE DUMMY ARGUMENT FOR IBCD.
0 90E6 2000 000E8 C0000    70          MVC DOPE(1),0(2)
0 90E6             000E8   71          LA 2,DOPE
0 90C6             000C8   72          ST 2,IBCD          STORING ADDRESS OF DOPE VECTOR
9 90C6       000C8         73          MVI IBCD,X'09'     SETTING THE TYPE CODE FOR THE IBCD ARGUMENT
C 90A6             000A8   74          BC 15,ENTSEQ
                           75 *
                           76 *  HAVE A 'SPECIAL' CALL
                           77 *
0 0001             00001   78 INTS     LA 2,1
C 9402             00404   79          ST 2,SPECFLG        SETTING THE SPECIAL FLAG ON.
                           80 *
                           81 *   IBCD  IS  A  SIMPLE  VARIABLE.
                           82 *
0 93FE             00400   83 NEXTY    LA 2,TPST           IBCD IS TREATED AS A
0 90C6             000C8   84          ST 2,IBCD           CALL-BY-VALUE ARGUMENT.
0 90E6             000E8   85          ST 2,DOPE
                           86 *
                           87 *  CHECKING FOR A 'SPECIAL' CALL
                           88 *
C 9402             00404   89          L 2,SPECFLG
2                          90          LTR 2,2
0 909A             0009C   91          BC 2,ISIMP          IT WAS A 'SPECIAL' CALL
                           92 *  DETERMINING TYPE OF THE IBCD ARGUMENT.
4 100C       0000C         93          TM 12(1),X'04'
0 90A0             000A2   94          BC 12,ISIMP1
4 90C6       000C8         95          MVI IBCD,X'84'     WAS REAL*4 TYPE
0 90A6             000A8   96          BC 15,ENTSEQ
2                          97 ISIMP    SR 2,2              RESETTING THE 'SPECIAL' CALL FLAG
C 9402             00404   98          ST 2,SPECFLG
2 90C6       000C8         99 ISIMP1   MVI IBCD,X'82'     WAS INTEGER*4 TYPE
                          100          DROP 9

                          102 *   THE ENTRY SEQUENCE CODE FOLLOWS
                          103 *
                          104          USING REG11,11
0                         105          CNOP 0,4
B D00C             0000C  106 ENTSEQ   STM 14,11,12(13)
0 C0F6             000F6  107          BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC ROUTINE
0E2E8D4C2D6D3             108 REG11    DC H'0',CL6'SYMBOL'
C09B0                     109          DC A(SAVE)
                          110 *  THE MODEL ARGUMENT LIST IS CREATED
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    20 SEP 73

)03D0                     111           DC X'84',AL3(XPAGE)
)03D4                     112           DC X'84',AL3(YPAGE)
)03D8                     113           DC X'84',AL3(HEIGHT)
)0000                     114 IBCD      DC X'00',AL3(0)          FILLED IN AT RUN TIME.
)03DC                     115           DC X'84',AL3(ANGLE)
)03E0                     116           DC X'82',AL3(NCHAR)
)0000                     117           DC X'10',AL3(0)
                          118 *
                          119 *  GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM.
                          120 *
) 8044          000F4     121           BAL 1,*+28
)03D0                     122           DC X'00',AL3(XPAGE)
)03D4                     123           DC X'00',AL3(YPAGE)
)03D8                     124           DC X'00',AL3(HEIGHT)
)0000                     125 DOPE      DC X'00',AL3(0)          FILLED IN AT RUN TIME
)03DC                     126           DC X'00',AL3(ANGLE)
)03E0                     127           DC X'80',AL3(NCHAR)

)                         129           BALR 8,0
                          130           USING *,8
) 8026          0011C     131           BC 15,SYMBOL2            BRANCHING AROUND OS-TYPE ENTRY
                          132 *  AN ENTRY POINT USING OS-TYPE LINKAGE CONVENTIONS.
: D00C          0000C     133 SYMB1     STM 14,12,12(13)         OS-TYPE ENTRY POINT.

) 0004          00004     135           LA 3,4                   READJUSTING THE BASE REGISTER CONTENTS
3                         136           SR 15,3                  REGISTER 15 MUST ALWAYS CONTAIN
=                         137           LR 8,15                  THE ADDRESS OF SYMB1.
) 88BA          C09B0     138           LA 9,SAVE                PERFORMING AN OS-TYPE SAVE OPERATION
) 0008          CC008     139           ST 9,8(13)
) 88BE          00984     140           ST 13,SAVE+4
9                         141           LR 13,9
) 0001          00001     142           LA 2,1                   SETTING THE ENTRY SEQUENCE FLAG TO ONE
) 88B6          009AC     143           ST 2,ENTFLG

) 832A          00420     145 SYMBOL2   LA    9,LINK         LOAD LINK FOR PLOT     * R7 - NO OF CHAR
) 831A          00410     146           L     10,ADPLOT      LOAD ADDRESS OF PLOT   * R8 BASE          SYMB0120
) C001          00001     147           LA    0,1            SET CONSTANT 1         * F0 - XC          SYMB0130
7 1000          00000     148           LM    2,7,0(1)       LOAD LINKAGE           * F2 - YC          SYMB0140
                          149 *         LTR   7,7            TEST FOR CORRECT       * F4 - SCRATCH     SYMB0150
                          150 *         BC    11,EREXIT       LINKAGE               * F6 - SCRATCH     SYMB0160
3 83F5     004EB          151           MVI   IC+3,3         IC=3                                      SYMB0170
) 834A          00440     152           LE    6,SEVEN        DIV=7                                     SYMB0180
C 7000          00000     153           L     7,0(0,7)       PICK UP N                                 SYMB0190
7                         154           LTR   7,7            IS N GREATER 0                            SYMB0200
) 8058          0014E     155           BC    13,NNTPOS       YES                    NO                 SYMB0210
= 5000     00000          156           TM    0(5),X'FF'     TEST FOR DOPE VECTOR    .                 SYMB0212
) 8C7A          00170     157           BC    7,NPOS          YES                   NO   .             SYMB0214
) 5000          00000     158           L     5,0(0,5)       LOAD ADD FROM VECTOR    .    .             SYMB0220
) 807A          00170     159           BC    15,NPOS         BRANCH  TO NPOS  .  .  .    .             SYMB0222
) 5003          00003     160 NNTPOS    LA    5,3(0,5)       BCD=BCD+3               . ... . .         SYMB0224
)                         161           AR    7,0             IS N LESS -1            .                 SYMB0230
) 8066          0015C     162           BC    11,NMTWO         YES      NO            .                 SYMB0240
2 83F5     004EB          163           MVI   IC+3,2           IC=2      .            .                 SYMB0250
)                         164 NMTWO     LR    7,0            N=1   . . .            .                 SYMB0260
L                         165           SR    1,1                                    .                 SYMB0270
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
0 5000             C0000   166          IC    1,0(0,5)       K=CONTENTS OF BCD     .                  SYMB0280
0 82D6             003CC   167          CH    1,THRTN        IS K GREATER 13       .                  SYMB0290
0 807A             00170   168          BC    2,NPOS          NO          YES      .                  SYMB0300
0 834E             00444   169          LE    6,FOUR          DIV=4       .        .                  SYMB0310
0 4000             00000   170 NPOS     LE    2,0(0,4)      PICK UP HGT  . . . . . .                SYMB0320
2                          171          LTER  2,2           IS HGT ZERO OR LESS                       SYMB0330
0 80DE             001D4   172          BC    13,HNEG        NO                              YES       SYMB0340
1 891A       00A10         173          MVI   STATE,1        STATE=1                         .         SYMB0350
6                          174          DER   2,6            FCT=HGT/DIV                     .         SYMB0360
0 6000             00000   175          LE    0,0(0,6)       PICK UP ANGLE                   .         SYMB0370
0 8366             0045C   176          CE    0,THETA        IS ANGLE=THETA                  .         SYMB0380
0 80A6             0019C   177          BC    7,STRTH         YES                      NO     .         SYMB0390
0 8362             00458   178          CE    2,FACT          IS FCT=FACT              .      .         SYMB0400
0 80DE             001D4   179          BC    8,HNEG           NO                      .      YES       SYMB0410
0 8362             00458   180          STE   2,FACT           FACT=FCT                .      .         SYMB0420
0 80D2             001C8   181          BC    15,GTCALC      GO TO CALCULATE OFFSETS   .      .         SYMB0430
0 8362             00458   182 STRTH    STE   2,FACT         FACT=FCT                  . . .  .         SYMB0440
0 8366             0045C   183          STE   0,THETA        THETA=ANGLE                     .         SYMB0450
0 8352             00448   184          ME    0,RADCO        CHANGE TO RADIANS               .         SYMB0460
0 836A             00460   185          STE   0,INCC                                        .         SYMB0470
0 8326             0041C   186          LA    1,ADANG                                       .         SYMB0480
0 831E             00414   187          L     15,ADSIN                                      .         SYMB0490
F                          188          BALR  14,15          CALL SIN                       .         SYMB0500
0 836E             00464   189          STE   0,INCS         INCS=SIN(THETA)                .         SYMB0510
0 8326             0041C   190          LA    1,ADANG                                       .         SYMB0520
0 8322             00418   191          L     15,ADCOS                                      .         SYMB0530
F                          192          BALR  14,15          CALL COS                       .         SYMB0540
0 836A             00460   193          STE   0,INCC         INCC=COS(THETA)                .         SYMB0550
3 8312             00408   194 GTCALC   STM   2,3,SVXYL                                     .         SYMB0560
0 82A6             0039C   195          BAL   4,CALC         CALCULATE OFFSETS              .         SYMB0570
3 8312             00408   196          LM    2,3,SVXYL                                     .         SYMB0580
0 2000             C0000   197 HNEG     LE    0,0(0,2)       PICK UP X                 . . .  .         SYMB0590
0 8356             0044C   198          CE    0,NNN          IS X EQUAL TO 999.0                     SYMB0600
0 80FA             001F0   199          BC    08,XNN          NO                 YES                  SYMB0610
                           200 *        CE    0,MZRO         IS X=-0.0           .                    SYMB0620
                           201 *        BC    8,XNN           NO                 YES                  SYMB0630
0 837A             00470   202          SE    0,XA+8          X=X-XA(2)+YA(2)    .                    SYMB0640
0 83AE             004A4   203          AE    0,YA+8                             .                    SYMB0650
0 83DA             004D0   204          STE   0,XO            XO=X               .                    SYMB0660
0 83E2             004D8   205          STE   0,XC            XC=X               .                    SYMB0670
0 3000             00000   206 XNN      LE    2,0(0,3)       PICK UP Y       . . .                   SYMB0680
0 8356             0044C   207          CE    2,NNN          IS Y EQUAL TO 999.0                     SYMB0690
0 8116             0020C   208          BC    08,YNN          NO                      YES             SYMB0700
                           209 *        CE    2,MZRO         IS Y=-0.0               .               SYMB0710
                           210 *        BC    8,YNN           NO                     YES             SYMB0720
0 837A             00470   211          SE    2,XA+8          Y=Y-XA(2)-YA(2)        .               SYMB0730
0 83AE             004A4   212          SE    2,YA+8                                 .               SYMB0740
0 83DE             004D4   213          STE   2,YO            YO=Y                   .               SYMB0750
0 83E6             004DC   214          STE   2,YC            YC=Y                   .               SYMB0760
0 83E2             004D8   215 YNN      LE    0,XC           X=XC                . . .               SYMB0770
0 83E6             004DC   216          LE    2,YC           Y=YC                                    SYMB0780
0 5000             C0000   217 NXTCH    IC    3,0(0,5)     * K=CONTENTS OF BCD                       SYMB0790
0 8336             0042C   218          N     3,CHMSK        K=K MODULO 128                          SYMB0800
0 0002             00002   219          SLL   3,2            SET UP K FOR INDEX                      SYMB0810
3 83F6             004EC   220          L     4,TABLE(3)     SET LOC TO ADDRESS OF OFFSETS           SYMB0820
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                    20 SEP 73
0 833A            00430   221          N     4,THBYT          FOR CHARACTER K          SYMB0830
6                         222          SR    6,6            SET J  TO NUMBER OF OFFSETS SYMB0840
3 83F7            004ED   223          IC    6,TABLE+1(3)     FOR CHARACTER K          SYMB0850
2                         224 NXTOF    SR    2,2          *                          SYMB0860
4 8000            00000   225          IC    2,0(4,8)       PICK UP OFFSETS FROM LOC SYMB0870
C 0004            00004   226          SRDL  2,4            SET I TO X-OFFSET        SYMB0880
0 82D6            003CC   227          CH    2,THRTN        IS I GREATER 13          SYMB0890
0 81C4            0028A   228          BC    10,SPCDE        NO                      SYMB0900
0 0002            00002   229          SLA   2,2            SET UP I FOR INDEX       SYMB0910
2 8372            00468   230          AE    0,XA(2)        X=X+XA(I)                SYMB0920
2 83A6            0049C   231          AE    2,YA(2)        Y=Y+YA(I)                SYMB0930
2                         232          SR    2,2                                     SYMB0940
0 0004            00004   233          SLDL  2,4            SET I TO Y-OFFSET        SYMB0950
C 0002            00002   234          SLA   2,2            SET UP I FOR INDEX       SYMB0960
2 83A6            0049C   235          SE    0,YA(2)        X=X-YA(I)                SYMB0970
2 8372            00468   236          AE    2,XA(2)        Y=Y-XA(I)                SYMB0980
0 83EA            004E0   237          STE   0,XT           XT=X                     SYMB0990
0 83EE            004E4   238          STE   2,YT           YT=Y                     SYMB1000
9                         239          LR    1,9                                     SYMB1010
A                         240          LR    15,10                                   SYMB1020
F                         241          BALR  14,15          CALL PLOT(XT,YT,IC)      SYMB1030
2 83F5      004EB         242          MVI   IC+3,2         IC=2                     SYMB1040
0 83E2            004D8   243          LE    0,XC           X=XC                     SYMB1050
0 83F6            004DC   244          LE    2,YC           Y=YC                     SYMB1060
0                         245 GTNOF    AR    4,0          * INCREASE LOC TO NEXT OFFSET PAIR SYMB1070
0 8138            0022E   246          BCT   6,NXTOF        J=J-1, IF J IS NOT ZERO REPEAT NXTOF SYMB1080
0 838E            00484   247          AE    0,XA+28        X=X+XA(7)                SYMB1090
0 83C2            004B8   248          AE    2,YA+28        Y=Y+YA(7)                SYMB1100
3 83F5      004EB         249 GTNCH    MVI   IC+3,3       * IC=3                     SYMB1110
0 83E2            004D8   250          STE   0,XC           XC=X                     SYMB1120
0 83E6            004DC   251          STE   2,YC           YC=Y                     SYMB1130
0                         252          AR    5,0            INCREASE BCD TO NEXT CHARACTER SYMB1140
0 811E            00214   253          BCT   7,NXTCH        N=N-1,IF N IS NOT ZERO REPEAT NXTCH SYMB1150
                          254 *
                          255 *   RETURNING  TO  THE  CALLING  PROGRAM
                          256 *
0 88B6            009AC   257 EREXIT   L 3,ENTFLG           LOAD THE ENTRY SEQUENCE FLAG
3                         258          LTR 3,3              TEST FOR ZERO
0 81BC            002B2   259          BC 8,WATRET          BRANCH TO WATFIV RETURN SEQUENCE
3                         260          SR 3,3
0 88B6            009AC   261          ST 3,ENTFLG          RESETTING FLAG TO ZERO
0 88BE            009B4   262          L 13,SAVE+4
C D00C            0000C   263          LM 14,12,12(13)
E                         264          BR 14                USING OS-TYPE RETURN
                          265 *   A  WATFIV  TYPE  RETURN
0 88BA            009B0   266 WATRET   LA 13,SAVE
0 C464            00464   267          BC 15,XRET(0,12)     BRANCH TO XRET RTN.
2                         268 SPCDE    SR    2,2          ** DECODE Y-OFFSET        SYMB1200
0 0004            00004   269          SLDA  2,4            SET I TO Y OFFSET        SYMB1210
0                         270          SR    2,0                                     SYMB1220
0 829E            00394   271          BC    4,PNUP                                  SYMB1230
0 824A            00340   272          BC    8,SUPSC        IF I=1, GO TO SUPERSCRIPT CODE SYMB1240
0                         273          SR    2,0                                     SYMB1250
0 820E            00304   274          BC    8,SUBSC        IF I=2, GO TO SUBSCRIPT CODE SYMB1260
0                         275          SR    2,0                                     SYMB1270
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
0 81F2             002E8   276          BC    8,CRET        IF I=3, GO TO CARRIAGE RETURN CODE      SYMB1280
0                          277          SR    2,0                                                   SYMB1290
C 8190             00286   278          BC    7,GTNCH       IF I=5, SKIP TO NEXT CHARACTER          SYMB1300
0 838E             00484   279 BCKSP    SE    0,XA+28    ** X=X-XA(7)                               SYMB1310
0 83C2             00488   280          SE    2,YA+28       Y=Y-YA(7)                               SYMB1320
C 8190             00286   281          BC    15,GTNCH      BRANCH TO GET NEXT CHARACTER            SYMB1330
0 83DA             004D0   282 CRET     LE    0,XO       **                                         SYMB1340
0 83DE             004D4   283          LE    2,YO                                                  SYMB1350
0 83D6             004CC   284          AE    0,YA+48       X                                       SYMB1360
0 83A2             00498   285          SE    2,XA+48                                               SYMB1370
0 83DA             004D0   286          STE   0,XO          XO=X                                    SYMB1380
0 83DE             004D4   287          STE   2,YO          YO=Y                                    SYMB1390
C 8190             00286   288          BC    15,GTNCH      BRANCH TO GET NEXT CHARACTER            SYMB1400
1 891A     00A10           289 SUBSC    CLI   STATE,1    ** TEST STATE                              SYMB1410
C 8190             00286   290          BC    4,GTNCH        IF 0, BRANCH TO GET NEXT CHARATER      SYMB1420
0 823A             00330   291          BC    8,SUBER        IF 1, BRANCH TO SUBER                  SYMB1430
1 891A     00A10           292          MVI   STATE,1        IF 2, RETURN TO NORMAL MODE,STATE=1    SYMB1440
0 8362             00458   293          LE    0,FACT                                                SYMB1450
0 835E             00454   294          DE    0,FCTR         FACT=FACT/FCTR                         SYMB1460
0 8362             00458   295          STE   0,FACT                                                SYMB1470
0 82AA             C03A0   296          BAL   4,CALCA        CALCULATE OFFSETS                      SYMB1480
0 83B6             004AC   297          AE    0,YA+16        X=X+YA(4)                              SYMB1490
0 8382             00478   298          SE    2,XA+16        Y=Y-XA(4)                              SYMB1500
C 8190             00286   299          BC    15,GTNCH       BRANCH TO GET NEXT CHARACTER           SYMB1510
0 891A     00A10           300 SUBER    MVI   STATE,0     * CHANGE TO SUBSCRIPT MODE,STATE=0       SYMB1520
0 83AE             0C4A4   301          AE    0,YA+8         X=X+YA(2)                              SYMB1530
0 837A             00470   302          SE    2,XA+8         Y=Y-XA(2)                              SYMB1540
0 8282             00378   303          BC    15,SUA         BRANCH                        . . . . . SYMB1550
1 891A     00A10           304 SUPSC    CLI   STATE,1    ** TEST STATE                          .   SYMB1560
0 8190             00286   305          BC    2,GTNCH        IF 2, BRANCH TO GET NEXT CHARACTER  .   SYMB1570
0 8276             0036C   306          BC    8,SUPER        IF 1, BRANCH TO SUPER               .   SYMB1580
1 891A     00A10           307          MVI   STATE,1        IF 0, RETURN TO NORMAL MODE,STATE=1 .   SYMB1590
0 8362             00458   308          LE    0,FACT                                            .   SYMB1600
0 835E             00454   309          DE    0,FCTR         FACT=FACT/FCTR                     .   SYMB1610
0 8362             00458   310          STE   0,FACT                                            .   SYMB1620
0 82AA             003A0   311          BAL   4,CALCA        CALCULATE OFFSETS                  .   SYMB1630
0 83AE             004A4   312          SE    0,YA+8         X=X-YA(2)                          .   SYMB1640
0 837A             C0470   313          AE    2,XA+8         Y=Y+XA(2)                          .   SYMB1650
C 8190             00286   314          BC    15,GTNCH       BRANCH TO GET NEXT CHARACTER       .   SYMB1660
2 891A     00A10           315 SUPER    MVI   STATE,2     * CHANGE TO SUPERSCRIPT MODE,STATE=2  .   SYMB1670
0 83B6             004AC   316          SE    0,YA+16        X=X-YA(4)                          .   SYMB1680
0 8382             00478   317          AE    2,XA+16        Y=Y+XA(4)                          .   SYMB1690
0 83E2             004D8   318 SUA      STE   0,XC          XC=X                          . . . . . SYMB1700
0 83E6             004DC   319          STE   2,YC          YC=Y                                    SYMB1710
0 8362             0C458   320          LE    0,FACT                                                SYMB1720
0 835F             00454   321          ME    0,FCTR         FACT=FACT*FCTR                         SYMB1730
0 8362             00458   322          STE   0,FACT                                                SYMB1740
0 82AA             003A0   323          BAL   4,CALCA        CALCULATE OFFSETS                      SYMB1750
C 8190             00286   324          BC    15,GTNCH       BRANCH TO GET NEXT CHARACTER           SYMB1760
3 83F5     004EB           325 PNUP     MVI   IC+3,3      ** RAISE PEN, IC=3                       SYMB1770
0 8182             00278   326          BC    15,GTNOF       BRANCH TO GET NEXT OFFSET              SYMB1780
0 8362             00458   327 CALC     LE    0,FACT      ** CALCULATE OFFSETS                       SYMB1790
0                          328 CALCA    LER   2,0                                                   SYMB1800
0 836A             00460   329          ME    0,INCC         X=FACT*INCC                            SYMB1810
0 836E             00464   330          ME    2,INCS         Y=FACT*INCS                            SYMB1820
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    SYMB1830
0                         331           LER   4,0             XI=X                                   SYMB1840
2                         332           LER   6,2             YI=Y                                   SYMB1850
3 833E                    333           LM    1,3,MFOR        I=1                                    SYMB1860
1 8372                    334 CALCB     STE   0,XA(1)         XA(I)=X                                SYMB1870
1 83A6                    335           STE   2,YA(1)         YA(I)=Y                                SYMB1880
4                         336           AER   0,4             X=X+XI                                 SYMB1890
6                         337           AER   2,6             Y=Y+YI                                 SYMB1900
2 82BC                    338           BXLE  1,2,CALCB       I=I+1 AND REPEAT UNTIL I IS 12         SYMB1910
0 83E2                    339           LE    0,XC            X=XC                                   SYMB1920
0 83E6                    340           LE    2,YC            Y=YC                                   SYMB1930
4                         341           BCR   15,4            RETURN                                 SYMB1940
D                         342 THRTN     DC    H'13'       ** CONSTANTS AND VARIABLES
                          343 XPAGE     DS F
                          344 YPAGE     DS F
                          345 HEIGHT    DS F
                          346 ANGLE     DS F
                          347 NCHAR     DS F
0                         348           CNOP 2,4
040404040                 349           DC CL6'    '  NO ARRAY NAME
0 C638                    350 STAR      BAL 15,XA1(0,12)   BRANCH TO XA1
00000                     351           DC AL1(0),AL3(0)
00000                     352           DC AL1(2),AL3(0)
00000                     353           DC A(0)
00400                     354           DC X'40',AL3(TPST)
                          355 TPST      DS F
                          356 SPECFLG   DS F
                          357 SVXYL     DS    2F                                                     SYMB1960
00000                     358 ADPLOT    DC   A(PLOT1)
00000                     359 ADSIN     DC    A(SIN1)
00000                     360 ADCOS     DC   A(COS1)
00460                     361 ADANG     DC    A(INCC)                                                SYMB2000
004E0                     362 LINK      DC    A(XT)                                                  SYMB2010
004E4                     363           DC    A(YT)                                                  SYMB2020
                          364           DC    X'80'                                                  SYMB2030
4E8                       365           DC    AL3(IC)                                                SYMB2040
0007F                     366 CHMSK     DC    F'127'                                                 SYMB2050
00FFF                     367 THBYT     DC    X'00000FFF'                                            SYMB2060
00004                     368 MFOR      DC    F'4'                                                   SYMB2070
00004                     369           DC    F'4'                                                   SYMB2080
00030                     370           DC    F'48'                                                  SYMB2090
00000                     371 SEVEN     DC    E'7.0'                                                 SYMB2100
00000                     372 FOUR      DC    E'4.0'                                                 SYMB2110
77D1D                     373 RADCO     DC    E'0.0174533'                                           SYMB2120
E7000                     374 NNN       DC    E'999.0'                                               SYMB2130
00000                     375 MZRO      DC    X'80000000'                                            SYMB2140
33333                     376 FCTR      DC    E'0.7'                                                 SYMB2150
00000                     377 FACT      DC    E'0.0'                                                 SYMB2160
00000                     378 THETA     DC    E'0.0'                                                 SYMB2170
00000                     379 INCC      DC    E'1.0'                                                 SYMB2180
00000                     380 INCS      DC    E'0.0'                                                 SYMB2190
000000000000              381 XA        DC    13F'0'                                                 SYMB2200
000000000000              382 YA        DC    13F'0'                                                 SYMB2210
00000                     383 XO        DC    F'0'                                                   SYMB2220
00000                     384 YO        DC    F'0'                                                   SYMB2230
00000                     385 XC        DC    F'0'                                                   SYMB2240
```

```
:CT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                                  SYMB2250
)0000                    386 YC       DC    F'0'                                                                 SYMB2260
:0000                    387 XT       DC    F'0'                                                                 SYMB2270
)0000                    388 YT       DC    F'0'                                                                 SYMB2280
:0000                    389 IC       DC    F'0'                                                                 SYMB2290
1 85F6      006EC        390 TABLE    MVI   S0,08      N=  0 HEX=00   CENTER SQUARE                              SYMB2300
: 85FD      006F3        391          MVI   S1,12      N=  1 HEX=01   CENTER OCTAGON                            SYMB2310
5 8608      C06FE        392          MVI   S2,06      N=  2 HEX=02   CENTER TRIANGLE                          SYMB2320
7 860D      00703        393          MVI   S3,07      N=  3 HEX=03   CENTER PLUS                               SYMB2330
7 8613      C0709        394          MVI   S4,07      N=  4 HEX=04   CENTER X                                  SYMB2340
7 8619      0070F        395          MVI   S5,07      N=  5 HEX=05   CENTER DIAMOND                          SYMB2350
7 861F      00715        396          MVI   S6,07      N=  6 HEX=06   CENTER UP ARROW                         SYMB2360
5 8625      C071B        397          MVI   S7,06      N=  7 HEX=07   CENTER BAR X                            SYMB2370
3 862A      C0720        398          MVI   S8,08      N=  8 HEX=08   CENTER Z                                 SYMB2380
7 8631      C0727        399          MVI   S9,07      N=  9 HEX=09   CENTER Y                                 SYMB2390
: 8637      C072D        400          MVI   SA,14      N= 10 HEX=0A   CENTER SQUARE X                         SYMB2400
) 860D      00703        401          MVI   S3,13      N= 11 HEX=0B   CENTER ASTERISK                         SYMB2410
5 8644      0073A        402          MVI   SC,06      N= 12 HEX=0C   CENTER DOUBLE BAR X                     SYMB2420
4 8649      0073F        403          MVI   SD,04      N= 13 HEX=0D   CENTER VERTICAL                         SYMB2430
5 87D0      008C6        404          MVI   SE,06      N= 14 HEX=0E   STAR                                     SYMB2440
2 8670      00766        405          MVI   SF,02      N= 15 HEX=0F   HORIZ VECTOR                            SYMB2450
2 864C      00742        406          MVI   SG,02      N= 16 HEX=10   VERTICAL VECTOR                          SYMB2460
1 8885      00978        407          MVI   U4,01      N= 17 HEX=11   ** BACKSPACE                            SYMB2470
3 87E8      008Dc        408          MVI   SL,03      N= 18 HEX=12   CARAT                                    SYMB2480
3 864E      00744        409          MVI   SJ,08      N= 19 HEX=13   EQUIVELENCE                             SYMB2490
5 87E3      C08D9        410          MVI   SK,05      N= 20 HEX=14   RIGHT ARROW                             SYMB2500
1 8884      C097A        411          MVI   U3,01      N= 21 HEX=15   ** CARRIAGE RETURN                      SYMB2510
3 865C      00752        412          MVI   SM,08      N= 22 HEX=16   NOT EQUAL                               SYMB2520
3 8654      0074A        413          MVI   SN,08      N= 23 HEX=17   PLUS MINUS                              SYMB2530
2 87EB      008E1        414          MVI   SO,02      N= 24 HEX=18   UNDERSCORE                              SYMB2540
1 8886      C097C        415          MVI   U5,01      N= 25 HEX=19   ** NULL CHARACTER                       SYMB2550
2 87ED      008E3        416          MVI   TL,02      N= 26 HEX=1A   OVERSCORE                              SYMB2560
3 881F      CC915        417          MVI   NH,08      N= 27 HEX=1B   INTEGRAL                               SYMB2570
5 882C      C0922        418          MVI   NJ,06      N= 28 HEX=1C   IMPLIES                                SYMB2580
3 8832      00928        419          MVI   NK,03      N= 29 HEX=1D   OR                                      SYMB2590
5 8835      C092B        420          MVI   NL,06      N= 30 HEX=1E                                           SYMB2600
) 8835      C092B        421          MVI   NL,13      N= 31 HEX=1F                                           SYMB2610
9 8842      00938        422          MVI   NM,09      N= 32 HEX=20   RIGHT BRACKET                          SYMB2620
9 884B      00941        423          MVI   NN,09      N= 33 HEX=21   LEFT  BRACKET                          SYMB2630
9 8854      C094A        424          MVI   NP,09      N= 34 HEX=22   MU                                      SYMB2640
5 885D      00953        425          MVI   NQ,06      N= 35 HEX=23   PI                                      SYMB2650
: 886B      C0961        426          MVI   NR,12      N= 36 HEX=24   PHI                                     SYMB2660
: 886E      CC964        427          MVI   NS,12      N= 37 HEX=25   THETA                                   SYMB2670
A 8863      00959        428          MVI   NT,10      N= 38 HEX=26   PSI                                     SYMB2680
7 887A      C0970        429          MVI   NU,07      N= 39 HEX=27   CHI                                     SYMB2690
9 8887      CC97D        430          MVI   NV,09      N= 40 HEX=28   OMEGA                                   SYMB2700
7 8890      00986        431          MVI   NW,07      N= 41 HEX=29   LAMBDA                                  SYMB2710
E 87BF      008B5        432          MVI   ND,14      N= 42 HEX=2A   ALPHA                                   SYMB2720
: 8897      C098D        433          MVI   NX,12      N= 43 HEX=2B   DELTA                                   SYMB2730
9 88A3      CC999        434          MVI   NY,09      N= 44 HEX=2C   EPSILON                                 SYMB2740
B 88AC      C09A2        435          MVI   NZ,08      N= 45 HEX=2D   ETA                                     SYMB2750
1 8882      CC978        436          MVI   U1,01      N= 46 HEX=2E   ** SUPERSCRIPT                          SYMB2760
1 8883      00979        437          MVI   U2,01      N= 47 HEX=2F   ** SUBSCRIPT                            SYMB2770
5 87FD      008F3        438          MVI   SV,05      N= 48 HEX=30   SUMATION                               SYMB2780
E 87EF      008E5        439          MVI   SW,14      N= 49 HEX=31   DIVIDE                                  SYMB2790
6 8664      0075A        440          MVI   SB,06      N= 50 HEX=32   LESS THAN OR EQUAL                      SYMB2800
```

ECT CODE	ADDR1 ADDR2	STMT	SOURCE STATEMENT			
5 8668	0075E	441	MVI	SX,06	N= 51 HEX=33	GREATER OR EQUAL	SYMB2800
4 8802	C08F8	442	MVI	SZ,04	N= 52 HEX=34	DELTA	SYMB2810
4 866E	00764	443	MVI	TO,04	N= 53 HEX=35	LEFT BRACE	SYMB2820
4 8670	00766	444	MVI	SF,04	N= 54 HEX=36	RIGHT BRACE	SYMB2830
2 8679	0076F	445	MVI	T3,02	N= 55 HEX=37	REVERSE SLASH	SYMB2840
7 88C6	008FC	446	MVI	T5,07	N= 56 HEX=38	GAMMA	SYMB2850
5 87D6	008CC	447	MVI	T6,05	N= 57 HEX=39	SQUARE ROOT	SYMB2860
A 880D	00903	448	MVI	T7,10	N= 58 HEX=3A		SYMB2870
D 880D	00903	449	MVI	T7,13	N= 59 HEX=3B		SYMB2880
5 881A	00910	450	MVI	T8,05	N= 60 HEX=3C	LEFT ARROW	SYMB2890
5 8827	C091D	451	MVI	NI,05	N= 61 HEX=3D	TIMES	SYMB2900
5 87DE	008D4	452	MVI	SI,05	N= 62 HEX=3E	UP ARROW	SYMB2910
5 87D8	008D1	453	MVI	SH,05	N= 63 HEX=3F	DN ARROW	SYMB2920
1 8881	C0977	454	MVI	U0,01	N= 64 HEX=40	BLANK	SYMB2930
9 867E	00774	455	MVI	TA,09	N= 65 HEX=41	A	SYMB2940
C 8687	00779	456	MVI	TB,12	N= 66 HEX=42	B	SYMB2950
8 8694	0078A	457	MVI	TC,08	N= 67 HEX=43	C	SYMB2960
7 868D	00783	458	MVI	TD,07	N= 68 HEX=44	D	SYMB2970
7 8672	0C763	459	MVI	T2,07	N= 69 HEX=45	E	SYMB2980
6 8672	00768	460	MVI	T2,06	N= 70 HEX=46	F	SYMB2990
A 8694	0078A	461	MVI	TC,10	N= 71 HEX=47	G	SYMB3000
6 86CE	00794	462	MVI	TH,06	N= 72 HEX=48	H	SYMB3010
6 86A4	C079A	463	MVI	TI,06	N= 73 HEX=49	I	SYMB3020
B 86AA	C07A0	464	MVI	TE,11	N= 74 HEX=4A	CENT	SYMB3030
5 86BB	0C7B1	465	MVI	TG,05	N= 75 HEX=4B	PERIOD	SYMB3040
3 8664	0075A	466	MVI	SB,03	N= 76 HEX=4C	LESS	SYMB3050
4 86C1	007B7	467	MVI	T9,04	N= 77 HEX=4D	LEFT PAREN	SYMB3060
5 8657	0074D	468	MVI	SS,05	N= 78 HEX=4E	PLUS	SYMB3070
2 86C5	007B8	469	MVI	SQ,02	N= 79 HEX=4F	VERTICAL	SYMB3080
A 86C7	C07BD	470	MVI	SR,10	N= 80 HEX=50	AMBERSEND	SYMB3090
5 86D2	007C9	471	MVI	TJ,05	N= 81 HEX=51	J	SYMB3100
7 86D7	007CD	472	MVI	TK,07	N= 82 HEX=52	K	SYMB3110
3 866F	00765	473	MVI	T1,03	N= 83 HEX=53	L	SYMB3120
5 86DE	007D4	474	MVI	TM,05	N= 84 HEX=54	M	SYMB3130
4 86E1	007D7	475	MVI	TN,04	N= 85 HEX=55	N	SYMB3140
C 86E8	007DE	476	MVI	TO,12	N= 86 HEX=56	O	SYMB3150
7 86F7	007ED	477	MVI	TP,07	N= 87 HEX=57	P	SYMB3160
C 86EB	007E1	478	MVI	TQ,12	N= 88 HEX=58	Q	SYMB3170
A 86F7	007ED	479	MVI	TP,10	N= 89 HEX=59	R	SYMB3180
7 8701	007F7	480	MVI	ST,07	N= 90 HEX=5A	EXCLAMATION	SYMB3190
C 8713	CC809	481	MVI	TR,12	N= 91 HEX=5B	DOLLAR SIGN	SYMB3200
8 871F	C0815	482	MVI	TX,08	N= 92 HEX=5C	ASTERISK	SYMB3210
4 8727	C081D	483	MVI	NA,04	N= 93 HEX=5D	RIGHT PARENTHESIS	SYMB3220
C 86B5	007AB	484	MVI	TF,12	N= 94 HEX=5E	SEMI COLON	SYMB3230
3 872B	00821	485	MVI	NB,03	N= 95 HEX=5F	NOT	SYMB3240
2 871F	00815	486	MVI	TX,02	N= 96 HEX=60	MINUS	SYMB3250
2 867C	00772	487	MVI	T4,02	N= 97 HEX=61	SLASH	SYMB3260
C 8730	00826	488	MVI	TS,12	N= 98 HEX=62	S	SYMB3270
4 8711	008C7	489	MVI	TT,04	N= 99 HEX=63	T	SYMB3280
6 86D1	007C7	490	MVI	TU,06	N=100 HEX=64	U	SYMB3290
3 873E	00834	491	MVI	TV,03	N=101 HEX=65	V	SYMB3300
5 86E3	007D9	492	MVI	TW,05	N=102 HEX=66	W	SYMB3310
5 8679	0076F	493	MVI	T3,05	N=103 HEX=67	X	SYMB3320
5 8741	CC837	494	MVI	TY,05	N=104 HEX=68	Y	SYMB3330
7 8746	0083C	495	MVI	TZ,07	N=105 HEX=69	Z	SYMB3340

```
11 878F  00885   496        MVI   NO,17      N=106 HEX=6A   INFINITY      SYMB3350
06 86BB  007B1   497        MVI   TG,06      N=107 HEX=6B   COMMA         SYMB3360
DE 874D  00843   498        MVI   NC,14      N=108 HEX=6C   PER CENT      SYMB3370
02 875B  00851   499        MVI   ND,02      N=109 HEX=6D   DASH          SYMB3380
03 866B  00761   500        MVI   SY,03      N=110 HEX=6E   GREATER THAN  SYMB3390
0D 8704  007FA   501        MVI   SU,13      N=111 HEX=6F   QUESTION MARK SYMB3400
09 86FB  007E1   502        MVI   TQ,09      N=112 HEX=70   0             SYMB3410
05 875D  00853   503        MVI   N1,05      N=113 HEX=71   1             SYMB3420
08 8762  00858   504        MVI   N2,08      N=114 HEX=72   2             SYMB3430
0D 876A  00860   505        MVI   N3,13      N=115 HEX=73   3             SYMB3440
08 877B  00871   506        MVI   N4,08      N=116 HEX=74   4             SYMB3450
09 8783  00879   507        MVI   N5,09      N=117 HEX=75   5             SYMB3460
0B 8770  00866   508        MVI   N6,11      N=118 HEX=76   6             SYMB3470
05 878A  00880   509        MVI   N7,05      N=119 HEX=77   7             SYMB3480
10 872E  00824   510        MVI   N8,16      N=120 HEX=78   8             SYMB3490
0C 878F  00885   511        MVI   N9,12      N=121 HEX=79   9             SYMB3500
0B 86B5  007AB   512        MVI   TF,11      N=122 HEX=7A   COLON         SYMB3510
0B 879B  00891   513        MVI   NE,11      N=123 HEX=7B   POUND         SYMB3520
10 87A6  0089C   514        MVI   NF,16      N=124 HEX=7C   AT            SYMB3530
04 87B6  008AC   515        MVI   NG,04      N=125 HEX=7D   APOSTROPE     SYMB3540
05 865C  00752   516        MVI   SM,05      N=126 HEX=7E   EQUAL         SYMB3550
09 87B6  008AC   517        MVI   NG,09      N=127 HEX=7F   QUOTATIONS    SYMB3560
240400404424     518 S0     DC    X'22240400404424'                       SYMB3570
2414C301103041   519 S1     DC    X'2224140301103041433424'               SYMB3580
24014124         520 S2     DC    X'2224014124'                           SYMB3590
2420220242       521 S3     DC    X'222420220242'                         SYMB3600
0440220044       522 S4     DC    X'220440220044'                         SYMB3610
2402204224       523 S5     DC    X'222402204224'                         SYMB3620
2024024224       524 S6     DC    X'222024024224'                         SYMB3630
20440440         525 S7     DC    X'2200440440'                           SYMB3640
40444C04000      526 S8     DC    X'22440444C04000'                       SYMB3650
0422442220       527 S9     DC    X'220422442220'                         SYMB3660
43313041311 00   528 SA     DC    X'224433130413110011314033 3'           SYMB3670
4044000          529 SC     DC    X'2244044000'                           SYMB3680
2420             530 SD     DC    X'222420'                               SYMB3690
26               531 SG     DC    X'2226'                                 SYMB3700
67F06525F0       532 SJ     DC    X'2767F06525F0'                         SYMB3710
23F0             533 SN     DC    X'2363F0'                               SYMB3720
3462666          534 SS     DC    X'4448462666'                           SYMB3730
4F06626F03357    535 SM     DC    X'2464F06626F03357'                     SYMB3740
2664F0           536 SB     DC    X'682664F0'                             SYMB3750
23F0             537 SX     DC    X'6323F0'                               SYMB3760
2628             538 SY     DC    X'246628'                               SYMB3770
                 539 T0     DC    X'69'                                   SYMB3780
                 540 T1     DC    X'29'                                   SYMB3790
2                541 SF     DC    X'2262'                                 SYMB3800
292656262262     542 T2     DC    X'69292656262262'                       SYMB3810
22F0             543 T3     DC    X'2962F0'                               SYMB3820
29               544 T4     DC    X'2269'                                 SYMB3830
25652528395968   545 TA     DC    X'2225652528395968 2'                   SYMB3840
2556265667       546 TB     DC    X'636556265667'                         SYMB3850
692922526368     547 TD     DC    X'68592922526368'                       SYMB3860
69392823325263   548 TC     DC    X'6859392823252636555'                  SYMB3870
2926666562       549 TH     DC    X'222926666962'                         SYMB3880
2242493959       550 TI     DC    X'3252424939 59'                        SYMB3890
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

7372624335364              551 TE       DC     X'6657372624335364F04248'          SYMB3900
5464535F0                  552 TF       DC     X'3536464535F0'                    SYMB3910
233434231                  553 TG       DC     X'423233434231'                    SYMB3920
85362                      554 T9       DC     X'69585362'                        SYMB3930
A                          555 SQ       DC     X'212A'                            SYMB3940
7384958252433              556 SR       DC     X'62373849582524334364'            SYMB3950
                           557 TU       CC     X'29'                              SYMB3960
2526369                    558 TJ       DC     X'2332526369'                      SYMB3970
22569F04762                559 TK       DC     X'29222569F04762'                  SYMB3980
945                        560 TM       DC     X'222945'                          SYMB3990
2                          561 TN       DC     X'6962'                            SYMB4000
2466269                    562 TW       DC     X'2922466269'                      SYMB4010
9F0                        563 TO       DC     X'4769F0'                          SYMB4020
9392823325263              564 TQ       DC     X'6859392823325263 68F04462'       SYMB4030
9596867562656              565 TP       DC     X'222959686756265665 62'           SYMB4040
4F0                        566 ST       DC     X'4944F0'                          SYMB4050
35232F0444656              567 SU       DC     X'32435232F04446566768593928'      SYMB4060
9                          568 TT       DC     X'2969'                            SYMB4070
2F02353646556              569 TR       DC     X'4942F02353646556636273868'       SYMB4080
5456327452367              570 TX       DC     X'25654553274523 67'               SYMB4090
83322                      571 NA       DC     X'29383322'                        SYMB4100
574                        572 NB       DC     X'257574'                          SYMB4110
7                          573 N8       DC     X'5667'                            SYMB4120
9392827365665              574 TS       DC     X'6859392827365656635232232536'    SYMB4130
269                        575 TV       DC     X'294269'                          SYMB4140
7424769                    576 TY       DC     X'2947424769'                      SYMB4150
92262F03556                577 TZ       DC     X'29692262F03656'                  SYMB4160
3293938F06922              578 NC       DC     X'3828293938F06922F05363625253'    SYMB4170
5                          579 ND       DC     X'1575'                            SYMB4180
9423252                    580 N1       DC     X'3849423252'                      SYMB4190
9596866242262              581 N2       DC     X'2839596866242262'                SYMB4200
95686756                   582 N3       DC     X'283959686756'                    SYMB4210
5656352322328              583 N6       DC     X'365656535232232839 5968'         SYMB4220
4645459524262              584 N4       DC     X'2924645459524262'                SYMB4230
25263655626                585 N5       DC     X'23325263655626'                  SYMB4240
9684342                    586 N7       DC     X'2969684342'                      SYMB4250
2526368593928              587 N9       DC     X'233252636859392826355566'        SYMB4260
4545357566626              588 NE       DC     X'24645453575666263353733'         SYMB4270
7473635445465              589 NF       DC     X'665747363544546567583827243353 64'  SYMB4280
35957F0372939              590 NG       DC     X'57495957F037293937'              SYMB4290
7463727161524              591 NO       DC     X'67574637271615 24'               SYMB4300
5464554647576              592          DC     X'344546455464757667'              SYMB4310
372268632                  593 SE       DC     X'325372268632'                    SYMB4320
4424A7A                    594 T6       DC     X'2334424A7A'                      SYMB4330
454                        595 SH       DC     X'423454'                          SYMB4340
9375749                    596 SI       DC     X'4249375749'                      SYMB4350
5565475                    597 SK       DC     X'1575565475'                      SYMB4360
562                        598 SL       DC     X'224662'                          SYMB4370
1                          599 SO       DC     X'1181'                            SYMB4380
\                          600 TL       DC     X'1A8A'                            SYMB4390
5F03334444333              601 SW       DC     X'1565F03334444333F03637474636'    SYMB4400
1562A7A                    602 SV       DC     X'7121562A7A'                      SYMB4410
77212                      603 SZ       DC     X'12477212'                        SYMB4420
94842485968                604 T5       DC     X'28394842485968'                  SYMB4430
4345444483858              605 T7       DC     X'42443454444838584849463656'      SYMB4440
```

```
ECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

6341575                    606 T8       DC    X'1536341575'                                                  SYMB4450
L3142495A6A79              607 NH       DC    X'122131424954A6A79'                                           SYMB4460
7F02763                    608 NI       DC    X'2367F02763'                                                  SYMB4470
465665727                  609 NJ       DC    X'245465665727'                                                SYMB4480
266                        610 NK       DC    X'264266'                                                      SYMB4490
535536374F076              611 NL       DC    X'142535536374F0766555372716'                                  SYMB4500
L424556474A3B              612 NM       DC    X'2131424556474A3B2B'                                          SYMB4510
L424536474A5B              613 NN       DC    X'61514245364 74A5B6B'                                         SYMB4520
7344353646764              614 NP       DC    X'2237344353646764 73'                                         SYMB4530
727675753                  615 NQ       DC    X'333727675753'                                                SYMB4540
53443545666F0              616 NT       DC    X'26363443545666F0'                                            SYMB4550
3F0                        617 NR       DC    X'4248F0'                                                      SYMB4560
5243353646657              618 NS       DC    X'372624335364665737F01575'                                    SYMB4570
5F026365363                619 NU       DC    X'2366F026365363'                                              SYMB4580
                           620 U0       DC    X'F0'                                                          SYMB4590
                           621 U1       DC    X'F1'       ENTER SUPERSCRIPT MODE OR LEAVE SUBSCRIPT.         SYMB4600
                           622 U2       DC    X'F2'       ENTER SUBSCRIPT MODE OR LEAVE SUPERSCRIPT.         SYMB4610
                           623 U3       DC    X'F3'       CARRIAGE RETURN.                                   SYMB4620
                           624 U4       DC    X'F4'       BACKSPACE CHARACTER.                               SYMB4630
                           625 U5       DC    X'F5'       NULL CHARACTER                                     SYMB4640
4334447445364              626 NV       DC    X'27243344474453 64'                                           SYMB4650
74563F02245                627 NW       DC    X'28374563F02245'                                              SYMB4660
3374656656453              628 NX       DC    X'584837465665645343343546'                                    SYMB4670
736344363F065              629 NY       DC    X'6747363443 63F06525'                                         SYMB4680
7463446576652              630 NZ       DC    X'2637463446576652'                                            SYMB4690
                           631          DS 0F
00000                      632 ENTFLG   DC F'0'
                           633 SAVE     DS 24F
                           634 STATE    DS    C                                                              SYMB4710
                           635          END                                                                    SYMB4720
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                            1 SIMBLE   START 0         CALL SIMBLE(X,Y,HGT,IBCD,TH,N)      V127
                            2 *
                            3 *   THIS SUBPROGRAM IS THE WATFIV IMPLEMENTATION VERSION OF THE CALCOMP
                            4 *   PLOT ROUTINE CALLED SIMBLE.  IT IS CALLED TO DRAW TEXT SUCH AS
                            5 *   TITLES AND TO DRAW SPECIAL CENTERED SYMBOLS.
                            6 *
                            7          EXTRN SIN1      SIN1 AND COS1 ARE THE SIN AND COS FUNCTIONS.
                            8          EXTRN COS1
                            9          EXTRN PLOT1     OS-TYPE ENTRY POINT TO PLOT ROUTINE.
                           10 *
                           11 *
                           12 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                           13 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                           14 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                           15 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                           16 *
                           17 XENTSPEC EQU 246
                           18 XRET     EQU 1124
                           19 XA1      EQU 1592
                           20 *
                           21 *
590                        22          BALR 9,0
                           23          USING *,9
                           24 *
                           25 *   CHECKING TYPE OF CALL MADE TO SIMBLE.
                           26 *
820 1014          00014    27          L 2,20(0,1)          LOAD ADDRESS OF ICODE
820 2000          00000    28          L 2,0(0,2)           OBTAIN CONTENTS OF ICODE
222 .                      29          LTR 2,2              IS IT A SPECIAL CALL
740 906C          0006E    30          BC 4,INTS            BRANCH IF SO
                           31 *   IT WAS A 'STANDARD' CALL.
                           32 *
                           33 *   DETERMINING WHETHER IBCD IS A SIMPLE VARIABLE, A VECTOR, OR A
                           34 *   HOLLERITH STRING.
                           35 *
109 100C    0000C          36          TM 12(1),X'09'       HOLLERITH STRING ?
710 9052          00054    37          BC 1,CHARK           YES
190 100C    0000C          38          TM 12(1),X'90'       VECTOR ?
7C0 9074          00076    39          BC 12,NEXTY          NO.  MUST BE SIMPLE VARIABLE
                           40 *
                           41 *   IBCD  IS  A  VECTOR
                           42 *
820 100C          0030C    43          L 2,12(0,1)  OBTAIN ADDRESS OF IBCD'S STAR ROUTINE
830 2004          00004    44          L 3,4(0,2)   OBTAIN ADDRESS OF THE CALLING VECTOR
820 200C          0000C    45          L 2,12(0,2)  OBTAIN THE VECTOR LENGTH FOR THE ACTUAL ARGUMENT
                           46 *  IT IS STORED IN A LOCATION REFERENCED BY THE STAR ROUTINE FOR THE
                           47 *  DUMMY ARGUMENT REPRESENTING IBCD.
020 9426          00428    48          ST 2,TPST            STORING THE VECTOR LENGTH
                           49 *  STORING THE VECTOR ADDRESS IN THE OS-TYPE ARGUMENT LIST.
030 90E6          000E8    50          ST 3,DOPE
120 9412          00414    51          LA 2,STAR            OBTAIN ADDRESS OF THE STAR RTN. FOR ARRAY
020 90C6          000C8    52          ST 2,IBCD            PUTTING THIS ADDRESS IN THE MODEL ARGUMENT
                           53 *                             LIST.
                           54 *
                           55 *  DETERMINING TYPE OF IBCD ARGUMENT
```

```
BJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

102 100C       0000C          56          TM 12(1),X'02'
710 904A             0C04C    57          BC 1,INTV
294 90C6       000C8          58          MVI IBCD,X'94'    IS REAL*4 TYPE
7F0 90A6             000A8    59          BC 15,ENTSEQ
292 90C6       000C8          60 INTV     MVI IBCD,X'92'    IS INTEGER*4 TYPE
7F0 90A6             000A8    61          BC 15,ENTSEQ

                              63 *
                              64 *  IBCD  IS  A  HOLLERITH  STRING.
                              65 *
820 100C             0030C    66 CHARK    L 2,12(0,1)  OBTAIN ADDRESS OF THE ACTUAL DOPE VECTOR FOR IBCD
                              67 *  MOVING THE VECTOR LENGTH INTO THE DOPE VECTOR LENGTH FIELD
                              68 *  OF THE DUMMY ARGUMENT FOR IBCD.
200 90E6 2000 00CE8 C0000     69          MVC DOPE(1),0(2)
120 90E6             000E8    70          LA 2,DOPE
020 90C6             000C8    71          ST 2,IBCD          STORING ADDRESS OF DOPE VECTOR
209 90C6       000C8          72          MVI IBCD,X'09'     SETTING THE TYPE CODE FOR THE IBCD ARGUMENT
7F0 90A6             000A8    73          BC 15,ENTSEQ
                              74 *
                              75 *  HAVE A 'SPECIAL' CALL
                              76 *
120 0001             00001    77 INTS     LA 2,1
020 942A             0042C    78          ST 2,SPECFLG        SETTING THE SPECIAL FLAG ON.
                              79 *
                              80 *   IBCD  IS  A  SIMPLE  VARIABLE.
                              81 *
120 9426             00428    82 NEXTY    LA 2,TPST           IBCD IS TREATED AS A
020 90C6             000C8    83          ST 2,IBCD           CALL-BY-VALUE ARGUMENT.
020 90E6             000E8    84          ST 2,DOPE
                              85 *
                              86 *  CHECKING FOR A 'SPECIAL' CALL
                              87 *
820 942A             0042C    88          L 2,SPECFLG
222                           89          LTR 2,2
720 909A             0009C    90          BC 2,ISIMP          IT WAS A 'SPECIAL' CALL
                              91 *  DETERMINING TYPE OF THE IBCD ARGUMENT.
104 100C       0000C          92          TM 12(1),X'04'
7CC 90A0             0C0A2    93          BC 12,ISIMP1
284 90C6       000C8          94          MVI IBCD,X'84'      WAS REAL*4 TYPE
7F0 90A6             000A8    95          BC 15,ENTSEQ
322                           96 ISIMP    SR 2,2              RESETTING THE 'SPECIAL' CALL FLAG
020 942A             0042C    97          ST 2,SPECFLG
282 90C6       000C8          98 ISIMP1   MVI IBCD,X'82'      WAS INTEGER*4 TYPE
                              99          DROP 9

                             101 *   THE ENTRY SEQUENCE CODE FOLLOWS
                             102 *
                             103          USING REG11,11
700                          104          CNOP 0,4
0EB D00C             0000C   105 ENTSEQ   STM 14,11,12(13)
5B0 C0F6             000F6   106          BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC RTN.
000E2C9E4C2D3C5              107 REG11    DC H'0',CL6'SIMBLE'
00C0390                      108          DC A(SAVE)
                             109 *  THE MODEL ARGUMENT LIST IS CREATED
40003F8                      110          DC X'84',AL3(XPAGE)
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

40003FC                    111          DC X'84',AL3(YPAGE)
4000400                    112          DC X'84',AL3(HEIGHT)
0003000                    113 IBCD     DC X'00',AL3(0)        FILLED IN AT RUN TIME.
4000404                    114          DC X'84',AL3(ANGLE)
2000408                    115          DC X'82',AL3(NCHAR)
0C000C0                    116          DC X'10',AL3(0)
                           117 *
                           118 *  GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM.
                           119 *
510 8044            000F4  120          BAL 1,*+28
00003F8                    121          DC X'00',AL3(XPAGE)
00003FC                    122          DC X'00',AL3(YPAGE)
0000400                    123          DC X'00',AL3(HEIGHT)
0000000                    124 DOPE     DC X'00',AL3(0)        FILLED IN AT RUN TIME
0000404                    125          DC X'00',AL3(ANGLE)
0000408                    126          DC X'80',AL3(NCHAR)

0580                       128          BALR 8,0
                           129          USING *,8
190 834A            00440  130          LA 9,LINK
8A0 833A            C0430  131          L     10,ADPLOT     LOAD ADDRESS OF PLOT   * R8 BASE          SYMB044D
100 0001            00001  132          LA    0,1           SET CONSTANT 1         * F0 - XC          SYMB044E
827 1000            00000  133          LM    2,7,0(1)      LOAD LINKAGE           * F2 - YC          SYMB044F
                           134 *        LTR   7,7           TEST FOR CORRECT       * F4 - SCRATCH     SYMB044G
                           135 *        BC    11,ERREXIT     LINKAGE               * F6 - SCRATCH     SYMB044H
                                                                                                       SYMB044I
203 8415      0050B        136          MVI   IC+3,3        IC=3                                       SYMB045
860 836A            00460  137          LE    6,SEVEN       DIV=7                                      SYMB045A
870 7C00            C0000  138          L     7,0(0,7)      PICK UP N                                  SYMB045B
277                        139          LTR   7,7           IS N GREATER 0                             SYMB045C
7D0 8032            00128  140          BC    13,NNTPOS      YES                        NO             SYMB045D
1FF 5000      C0000        141          TM    0(5),X'FF'    TEST FOR DOPE VECTOR        .              SYMB045E
770 8C54            0014A  142          BC    7,NPOS         YES                  NO     .              SYMB045F
850 5000            CC000  143          L     5,0(0,5)      LOAD ADD FROM VECTOR    .     .              SYMB045G
7F0 8054            0014A  144          BC    15,NPOS        BRANCH  TO NPOS  .  .  .     .              SYMB045H
15C 5003            00003  145 NNTPOS   LA    5,3(0,5)      BCD=BCD+3                .  ...  .  .       SYMB045I
A70                        146          AR    7,0            IS N LESS -1             .                 SYMB046
780 8040            00136  147          BC    11,NMTWO        YES       NO            .                 SYMB046A
202 8415      0050B        148          MVI   IC+3,2          IC=2       .            .                 SYMB046B
87C                        149 NMTWO    LR    7,0           N=1   .  .  .            .                 SYMB046C
B11                        150          SR    1,1                                    .                 SYMB046D
310 5000            00000  151          IC    1,0(0,5)       K=CONTENTS OF BCD       .                 SYMB046E
910 8296            0038C  152          CH    1,THRTN        IS K GREATER 13         .                 SYMB046F
720 8C54            0014A  153          BC    2,NPOS          NO          YES        .                 SYMB046G
860 836E            00464  154          LE    6,FOUR          DIV=4        .         .                 SYMB046H
82C 4000            00000  155 NPOS     LE    2,0(0,4)      PICK UP HGT  .  .  .  .  .  .             SYMB046I
1222                       156          LTER  2,2           IS HGT ZERO OR LESS                        SYMB047
7D0 8088            001AE  157          BC    13,HNEG        NO                             YES        SYMB047A
1201 8ABA     00BB0        158          MVI   STATE,1        STATE=1                         .         SYMB047B
1D26                       159          DER   2,6           FCT=HGT/DIV                     .         SYMB047C
7800 6000           00000  160          LE    0,0(0,6)      PICK UP ANGLE                   .         SYMB047D
7900 8386           0047C  161          CE    0,THETA       IS ANGLE=THETA                  .         SYMB047E
,770 8080           00176  162          BC    7,STRTH        YES                  NO        .         SYMB047F
7920 8382           00478  163          CE    2,FACT          IS FCT=FACT         .         .         SYMB047G
,780 8088           001AE  164          BC    8,HNEG           NO                 .         YES       SYMB047H
7020 8382           00478  165          STE   2,FACT           FACT=FCT           .         .
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

7F0 80AC            001A2   166          BC    15,GTCALC      GO TO CALCULATE OFFSETS .    .          SYMB047I
D20 8382            00478   167 STRTH    STE   2,FACT         FACT=FCT             . . .   .          SYMB048
D00 8386            0047C   168          STE   0,THETA        THETA=ANGLE                  .          SYMB048A
C00 8372            00468   169          ME    0,RADCO        CHANGE TO RADIANS            .          SYMB048B
D00 838A            00480   170          STE   0,INCC                                      .          SYMB048C
110 8346            0043C   171          LA    1,ADANG                                     .          SYMB048D
3F0 833E            00434   172          L     15,ADSIN                                    .          SYMB048E
5EF                 00434   173          BALR  14,15          CALL SIN                     .          SYMB048F
D00 838E            00484   174          STE   0,INCS         INCS=SIN(THETA)              .          SYMB048G
110 8346            0043C   175          LA    1,ADANG                                     .          SYMB048H
3F0 8342            00438   176          L     15,ADCOS                                    .          SYMB048I
5EF                 00438   177          BALR  14,15          CALL COS                     .          SYMB049
D00 838A            00480   178          STE   0,INCC         INCC=COS(THETA)              .          SYMB049A
D23 82FA            C03F0   179 GTCALC   STM   2,3,SVXYL                                   .          SYMB049B
540 8266            0035C   180          BAL   4,CALC         CALCULATE OFFSETS            .          SYMB049C
823 82FA            003F0   181          LM    2,3,SVXYL                                   .          SYMB049D
800 2000            C0000   182 HNEG     LE    0,0(0,2)       PICK UP X                . . .      SYMB049E
900 8376            0046C   183          CE    0,NNN          IS X EQUAL TO 999.0                     SYMB049F
780 80D4            001CA   184          BC    08,XNN          NO                YES                   SYMB049G
                            185 *        CE    0,MZRO         IS X=-0.0          .                    SYMB049H
                            186 *        BC    8,XNN           NO                YES                   SYMB049I
D00 839A            0C490   187          SE    0,XA+8          X=X-XA(2)+YA(2)    .                    SYMB050
A00 83CE            0C4C4   188          AE    0,YA+8                             .                    SYMB050A
D00 83FA            C04F0   189          STE   0,XO            XO=X               .                    SYMB050B
D00 8402            0C4F8   190          STE   0,XC            XC=X               .                    SYMB050C
820 3000            C0000   191 XNN      LE    2,0(0,3)       PICK UP Y       . . .                   SYMB050D
D20 8376            0046C   192          CE    2,NNN          IS Y EQUAL TO 999.0                     SYMB050E
780 80F0            001E6   193          BC    08,YNN          NO                     YES              SYMB050F
                            194 *        CE    2,MZRO         IS Y=-0.0               .                SYMB050G
                            195 *        BC    8,YNN           NO                     YES              SYMB050H
820 839A            00490   196          SE    2,XA+8          Y=Y-XA(2)-YA(2)         .               SYMB050I
820 83CE            004C4   197          SE    2,YA+8                                  .               SYMB051
D20 83FE            0C4F4   198          STE   2,YO            YO=Y                    .               SYMB051A
820 8406            004FC   199          STE   2,YC            YC=Y                    .               SYMB051B
D00 8402            004F8   200 YNN      LE    0,XC           X=XC                 . . .              SYMB051C
820 8406            0C4FC   201          LE    2,YC           Y=YC                                    SYMB051D
830 5000            C0000   202 NXTCH    IC    3,0(0,5)     * K=CONTENTS OF BCD                       SYMB051E
830 8356            0044C   203          N     3,CHMSK        K=K MODULO 128                          SYMB051F
D30 0002            00002   204          SLL   3,2            SET UP K FOR INDEX                      SYMB051G
143 8416            0050C   205          L     4,TABLE(3)     SET LOC TO ADDRESS OF OFFSETS           SYMB051H
440 835A            00450   206          N     4,THBYT          FOR CHARACTER K                       SYMB051I
166                 00450   207          SR    6,6            SET J  TO NUMBER OF OFFSETS             SYMB052
163 8417            0050D   208          IC    6,TABLE+1(3)     FOR CHARACTER K                       SYMB052A
122                 00450   209 NXTOF    SR    2,2          *                                         SYMB052B
824 80C0            C0000   210          IC    2,0(4,8)       PICK UP OFFSETS FROM LOC                SYMB052C
120 0004            00004   211          SRDL  2,4            SET I TO X-OFFSET                       SYMB052D
D20 8296            0C38C   212          CH    2,THRTN        IS I GREATER 13                         SYMB052E
7A0 8184            0027A   213          BC    10,SPCDE        NO                                     SYMB052F
120 0002            00002   214          SLA   2,2            SET UP I FOR INDEX                      SYMB052G
102 8392            0C488   215          AE    0,XA(2)        X=X+XA(I)                               SYMB052H
122 83C6            0048C   216          AE    2,YA(2)        Y=Y+YA(I)                               SYMB052I
122                 00450   217          SR    2,2                                                    SYMB053
120 C004            0C004   218          SLDL  2,4            SET I TO Y-OFFSET                       SYMB053A
120 C002            00002   219          SLA   2,2            SET UP I FOR INDEX                      SYMB053B
102 83C6            0C48C   220          SE    0,YA(2)        X=X-YA(I)                               SYMB053C
```

```
BJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
A22 8392             00488   221           AE    2,XA(2)         Y=Y-XA(I)                                SYMB053D
D00 840A             00500   222           STE   0,XT            XT=X                                     SYMB053E
D20 840E             00504   223           STE   2,YT            YT=Y                                     SYMB053F
819                          224           LR    1,9                                                      SYMB053G
3FA                          225           LR    15,10                                                    SYMB053H
5EF                          226           BALR  14,15           CALL PLOT(XT,YT,IC)                      SYMB053I
202 8415       00508         227           MVI   IC+3,2          IC=2                                     SYMB054
800 8402             004F8   228           LE    0,XC            X=XC                                     SYMB054A
820 8406             004FC   229           LE    2,YC            Y=YC                                     SYMB054B
A40                          230 GTNOF     AR    4,0           * INCREASE LOC TO NEXT OFFSET PAIR        SYMB054C
560 8112             00208   231           BCT   6,NXTOF         J=J-1, IF J IS NOT ZERO REPEAT NXTOF     SYMB054D
A00 83AE             004A4   232           AE    0,XA+28         X=X+XA(7)                                SYMB054E
A20 83E2             004D8   233           AE    2,YA+28         Y=Y+YA(7)                                SYMB054F
203 E415       00508         234 GTNCH     MVI   IC+3,3        * IC=3                                     SYMB054G
D00 8402             004F8   235           STE   0,XC            XC=X                                     SYMB054H
D20 84C6             004FC   236           STE   2,YC            YC=Y                                     SYMB054I
A50                          237           AR    5,0             INCREASE BCD TO NEXT CHARACTER           SYMB055
670 80F8             001EE   238           BCT   7,NXTCH         N=N-1,IF N IS NOT ZERO REPEAT NXTCH      SYMB055A
                             239 *
                             240 *     RETURNING TO THE CALLING PROGRAM
                             241 *
1DC 829A             00390   242 EREXIT    LA 13,SAVE
7F0 C464             00464   243           BC 15,XRET(0,12)
322                          244 SPCDE     SR    2,2           ** DECODE Y-OFFSET                       SYMB055F
F20 0004             00004   245           SLDA  2,4             SET I TO Y OFFSET                        SYMB055G
320                          246           SR    2,0                                                      SYMB055H
740 825E             00354   247           BC    4,PNUP                                                   SYMB055I
780 820A             00300   248           BC    8,SUPSC         IF I=1, GO TO SUPERSCRIPT CODE           SYMB056
320                          249           SR    2,0                                                      SYMB056A
780 81CE             002C4   250           BC    8,SUBSC         IF I=2, GO TO SUBSCRIPT CODE             SYMB056B
320                          251           SR    2,0                                                      SYMB056C
78C 81B2             002A8   252           BC    8,CRET          IF I=3, GO TO CARRIAGE RETURN CODE       SYMB056D
320                          253           SR    2,0                                                      SYMB056E
770 816A             00260   254           BC    7,GTNCH         IF I=5, SKIP TO NEXT CHARACTER           SYMB056F
B00 83AE             004A4   255 BCKSP     SE    0,XA+28       ** X=X-XA(7)                             SYMB056G
B20 83E2             004D8   256           SE    2,YA+28         Y=Y-YA(7)                                SYMB056H
7F0 816A             00260   257           BC    15,GTNCH        BRANCH TO GET NEXT CHARACTER             SYMB056I
B00 83FA             004F0   258 CRET      LE    0,XO          **                                         SYMB057
B20 83FE             004F4   259           LE    2,YO                                                     SYMB057A
A00 83F6             004EC   260           AE    0,YA+48         X                                        SYMB057B
320 83C2             004B8   261           SE    2,XA+48                                                  SYMB057C
D00 83FA             004F0   262           STE   0,XO            XO=X                                     SYMB057D
D20 83FE             004F4   263           STE   2,YO            YO=Y                                     SYMB057E
7F0 816A             00260   264           BC    15,GTNCH        BRANCH TO GET NEXT CHARACTER             SYMB057F
501 8ABA       00BB0         265 SUBSC     CLI   STATE,1       ** TEST STATE                            SYMB057G
740 816A             00260   266           BC    4,GTNCH          IF 0, BRANCH TO GET NEXT CHARATER       SYMB057H
780 81FA             002F0   267           BC    8,SUBER          IF 1, BRANCH TO SUBER                   SYMB057I
201 8ABA       00BB0         268           MVI   STATE,1          IF 2, RETURN TO NORMAL MODE,STATE=1     SYMB058
B00 8382             00478   269           LE    0,FACT                                                   SYMB058A
D00 837E             00474   270           DE    0,FCTR           FACT=FACT/FCTR                          SYMB058B
D00 8382             00478   271           STE   0,FACT                                                   SYMB058C
540 826A             00360   272           BAL   4,CALCA        CALCULATE OFFSETS                         SYMB058D
A00 83D6             004CC   273           AE    0,YA+16         X=X+YA(4)                                SYMB058E
320 83A2             00498   274           SE    2,XA+16         Y=Y-XA(4)                                SYMB058F
7F0 816A             00260   275           BC    15,GTNCH        BRANCH TO GET NEXT CHARACTER             SYMB058G
```

```
JECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    SYMB058H
00 8ABA       00BB0          276 SUBER    MVI   STATE,0      * CHANGE TO SUBSCRIPT MODE,STATE=0       SYMB058I
00 83CE             004C4    277          AE    0,YA+8         X=X+YA(2)                              SYMB059
20 839A             CC490    278          SE    2,XA+8         Y=Y-XA(2)                              SYMB059A
F0 8242             00338    279          BC    15,SUA         BRANCH                 . . . . . .     SYMB059B
01 8ABA       00BB0          280 SUPSC    CLI   STATE,1    ** TEST STATE                            .  SYMB059C
20 816A             00260    281          BC    2,GTNCH        IF 2, BRANCH TO GET NEXT CHARACTER     .  SYMB059D
80 8236             0032C    282          BC    8,SUPER        IF 1, BRANCH TO SUPER                  .  SYMB059E
01 8ABA       00BB0          283          MVI   STATE,1        IF 0, RETURN TO NORMAL MODE,STATE=1    .  SYMB059F
00 8382             00478    284          LE    0,FACT                                                .  SYMB059G
00 837E             00474    285          DE    0,FCTR         FACT=FACT/FCTR                         .  SYMB059H
00 8382             00478    286          STE   0,FACT                                                .  SYMB059I
40 826A             00360    287          BAL   4,CALCA        CALCULATE OFFSETS                      .  SYMB060
00 83CE             004C4    288          SE    0,YA+8         X=X-YA(2)                              .  SYMB060A
2C 839A             CC490    289          AE    2,XA+8         Y=Y+XA(2)                              .  SYMB060B
F0 816A             00260    290          BC    15,GTNCH       BRANCH TO GET NEXT CHARACTER           .  SYMB060C
02 8ABA       00BB0          291 SUPER    MVI   STATE,2      * CHANGE TO SUPERSCRIPT MODE,STATE=2     .  SYMB060D
00 83D6             004CC    292          SE    0,YA+16        X=X-YA(4)                              .  SYMB060E
20 83A2             00498    293          AE    2,XA+16        Y=Y+XA(4)                              .  SYMB060F
00 8402             004F8    294 SUA      STE   0,XC           XC=X                   . . . . . .     SYMB060G
20 8406             004FC    295          STE   2,YC           YC=Y                                   SYMB060H
00 8382             00478    296          LE    0,FACT                                                SYMB060I
00 837E             00474    297          ME    0,FCTR         FACT=FACT*FCTR                         SYMB061
00 8382             00478    298          STE   0,FACT                                                SYMB061A
40 826A             00360    299          BAL   4,CALCA        CALCULATE OFFSETS                      SYMB061B
F0 816A             00260    300          BC    15,GTNCH       BRANCH TO GET NEXT CHARACTER           SYMB061C
03 8415       0050B          301 PNUP     MVI   IC+3,3     ** RAISE PEN, IC=3                       SYMB061D
F0 815C             00252    302          BC    15,GTNOF       BRANCH TO GET NEXT OFFSET              SYMB061E
00 8382             00478    303 CALC     LE    0,FACT     ** CALCULATE OFFSETS                     SYMB061F
20                           304 CALCA    LER   2,0                                                   SYMB061G
00 838A             CC480    305          ME    0,INCC         X=FACT*INCC                            SYMB061H
20 838E             00484    306          ME    2,INCS         Y=FACT*INCS                            SYMB061I
40                           307          LER   4,0            XI=X                                   SYMB062
62                           308          LER   6,2            YI=Y                                   SYMB062A
13 835E             00454    309          LM    1,3,MFOR       I=1                                    SYMB062B
01 8392             CC488    310 CALCB    STE   0,XA(1)        XA(I)=X                . . . .         SYMB062C
21 83C6             004BC    311          STE   2,YA(1)        YA(I)=Y                        .        SYMB062D
04                           312          AER   0,4            X=X+XI                         .        SYMB062E
26                           313          AER   2,6            Y=Y+YI                         .        SYMB062F
12 827C             00372    314          BXLE  1,2,CALCB      I=I+1 AND REPEAT UNTIL I IS 12 . .      SYMB062G
00 8402             004F8    315          LE    0,XC           X=XC                                   SYMB062H
20 8406             004FC    316          LE    2,YC           Y=YC                                   SYMB062I
F4                           317          BCR   15,4           RETURN                                 SYMB063
0D                           318 THRTN    DC    H'13'      ** CONSTANTS AND VARIABLES               SYMB063B
                             319 SAVE     DS    24F                                                   
                             320 SVXYL    DS    2F                                                    
                             321 XPAGE    DS F                                                        
                             322 YPAGE    DS F                                                        
                             323 HEIGHT   DS F                                                        
                             324 ANGLE    DS F                                                        
                             325 NCHAR    DS F                                                        
00                           326          CNOP 2,4                                                    
4040404040                   327          DC CL6'     '  NO ARRAY NAME                               
F0 C638             00638    328 STAR     BAL 15,XA1(0,12)   BRANCH TO XA1                           
0C0000                       329          DC AL1(0),AL3(0)                                            
000000                       330          DC AL1(2),AL3(0)                                            
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
)000000                    331           DC A(0)
)000428                    332           DC X'40',AL3(TPST)
                           333 TPST      DS F
                           334 SPECFLG   DS F
)0C0000                    335 ADPLOT    DC    A(PLOT1)
)0C0000                    336 ADSIN     DC    A(SIN1)
)0C0000                    337 ADCOS     DC    A(COS1)
)0C0480                    338 ADANG     DC    A(INCC)
)000500                    339 LINK      DC    A(XT)
)000504                    340           DC    A(YT)
)                          341           DC    X'80'
)0508                      342           DC    AL3(IC)
)000007F                   343 CHMSK     DC    F'127'
)000FFF                    344 THBYT     DC    X'00000FFF'
)000004                    345 MFOR      DC    F'4'
)000004                    346           DC    F'4'
)000030                    347           DC    F'48'
170000                     348 SEVEN     DC    E'7.0'
140000                     349 FOUR      DC    E'4.0'
477D1D                     350 RADCO     DC    E'0.0174533'
33E7000                    351 NNN       DC    E'999.0'
)0C0000                    352 MZRO      DC    X'8C000000'
)B33333                    353 FCTR      DC    E'0.7'
)0C0000                    354 FACT      DC    E'0.0'
)0C)C00                    355 THETA     DC    E'0.0'
1100000                    356 INCC      DC    E'1.0'
)000000                    357 INCS      DC    E'0.0'
)0C)C0000C000000           358 XA        DC    13F'0'
)0C000CCCC00000            359 YA        DC    13F'0'
)000000                    360 XO        DC    F'0'
)0C)000                    361 YO        DC    F'0'
)0C0000                    362 XC        DC    F'0'
)0C0000                    363 YC        DC    F'0'
)000000                    364 XT        DC    F'0'
)0C0000                    365 YT        DC    F'0'
)000000                    366 IC        DC    F'0'
208 8616      0070C        367 TABLE     MVI   S0,08     N=  0 HEX=00   CENTER SQUARE
20C 861D      00713        368           MVI   S1,12     N=  1 HEX=01   CENTER OCTAGON
2C6 8628      0071E        369           MVI   S2,06     N=  2 HEX=02   CENTER TRIANGLE
207 862D      00723        370           MVI   S3,07     N=  3 HEX=03   CENTER PLUS
207 8633      00729        371           MVI   S4,07     N=  4 HEX=04   CENTER X
207 8639      0072F        372           MVI   S5,07     N=  5 HEX=05   CENTER DIAMOND
207 863F      00735        373           MVI   S6,07     N=  6 HEX=06   CENTER UP ARROW
206 8645      0073B        374           MVI   S7,06     N=  7 HEX=07   CENTER BAR X
208 864A      C0740        375           MVI   S8,08     N=  8 HEX=08   CENTER Z
207 8651      C0747        376           MVI   S9,07     N=  9 HEX=09   CENTER Y
20E 8657      0074D        377           MVI   SA,14     N= 10 HEX=0A   CENTER SQUARE X
20D 862D      C0723        378           MVI   S3,13     N= 11 HEX=0B   CENTER ASTERISK
206 8664      0075A        379           MVI   SC,06     N= 12 HEX=0C   CENTER DOUBLE BAR X
204 8669      0075F        380           MVI   SD,04     N= 13 HEX=0D   CENTER VERTICAL
206 89CC      00AC2        381           MVI   SE,06     N= 14 HEX=0E   STAR
202 8690      00786        382           MVI   SF,02     N= 15 HEX=0F   HORIZ VECTOR
202 866C      00762        383           MVI   SG,02     N= 16 HEX=10   VERTICAL VECTOR
201 8A8B      00B81        384           MVI   U4,01     N= 17 HEX=11   ** BACKSPACE
203 89E4      00ADA        385           MVI   SL,03     N= 18 HEX=12   CARAT
```

SYMB063F
SYMB063G
SYMB063H
SYMB063I
SYMB064
SYMB064A
SYMB064B
SYMB064C
SYMB064D
SYMB064E
SYMB064F
SYMB064G
SYMB064H
SYMB064I
SYMB065
SYMB065A
SYMB065B
SYMB065C
SYMB065D
SYMB065E
SYMB065F
SYMB065G
SYMB065H
SYMB065I
SYMB066
SYMB066A
SYMB066B
SYMB066C
SYMB066D
SYMB066E
SYMB066F
SYMB066G
SYMB066H
SYMB066I
SYMB067
SYMB067A
SYMB067B
SYMB067C
SYMB067D
SYMB067E
SYMB067F
SYMB067G
SYMB067H
SYMB067I
SYMB068
SYMB068A
SYMB068B
SYMB068C

BJECT CODE	ADDR1	ADDR2	STMT	SOURCE STATEMENT				
208 866E	00764		386	MVI	SJ,08	N= 19 HEX=13	EQUIVELENCE	SYMB068D
205 89DF	00AD5		387	MVI	SK,05	N= 20 HEX=14	RIGHT ARROW	SYMB068E
201 8A8A	C0880		388	MVI	U3,01	N= 21 HEX=15	** CARRIAGE RETURN	SYMB068F
20E 867C	C0772		389	MVI	SM,08	N= 22 HEX=16	NOT EQUAL	SYMB068G
208 8674	00768		390	MVI	SN,08	N= 23 HEX=17	PLUS MINUS	SYMB068H
202 89E7	00AC0		391	MVI	SO,02	N= 24 HEX=18	UNDERSCORE	SYMB068I
201 8A8C	00B82		392	MVI	U5,01	N= 25 HEX=19	** NULL CHARACTER	SYMB069
202 89E9	00ADF		393	MVI	TL,02	N= 26 HEX=1A	OVERSCORE	SYMB069A
208 8A1B	00B11		394	MVI	NH,08	N= 27 HEX=1B	INTEGRAL	SYMB069B
206 8A32	C0B28		395	MVI	NJ,06	N= 28 HEX=1C	IMPLIES	SYMB069C
203 8A38	00B2E		396	MVI	NK,03	N= 29 HEX=1D	OR	SYMB069D
206 8A3B	C0B31		397	MVI	NL,06	N= 30 HEX=1E		SYMB069E
20D 8A3B	C0B31		398	MVI	NL,13	N= 31 HEX=1F		SYMB069F
209 8A48	00B3E		399	MVI	NM,09	N= 32 HEX=20	RIGHT BRACKET	SYMB069G
209 8A51	00B47		400	MVI	NN,09	N= 33 HEX=21	LEFT BRACKET	SYMB069H
209 8A5A	C0B50		401	MVI	NP,09	N= 34 HEX=22	MU	SYMB069I
206 8A63	C0B59		402	MVI	NQ,06	N= 35 HEX=23	PI	SYMB070
20C 8A71	00B67		403	MVI	NR,12	N= 36 HEX=24	PHI	SYMB070A
20C 8A74	00B6A		404	MVI	NS,12	N= 37 HEX=25	THETA	SYMB070B
20A 8A69	00B5F		405	MVI	NT,10	N= 38 HEX=26	PSI	SYMB070C
207 8A80	00B76		406	MVI	NU,07	N= 39 HEX=27	CHI	SYMB070D
209 8A8D	00B83		407	MVI	NV,09	N= 40 HEX=28	OMEGA	SYMB070E
207 8A96	00B8C		408	MVI	NW,07	N= 41 HEX=29	LAMBDA	SYMB070F
20E 89BB	00AB1		409	MVI	NO,14	N= 42 HEX=2A	ALPHA	SYMB070G
20C 8A9D	00B93		410	MVI	NX,12	N= 43 HEX=2B	DELTA	SYMB070H
209 8AA9	00B9F		411	MVI	NY,09	N= 44 HEX=2C	EPSILON	SYMB070I
208 8AB2	00BA8		412	MVI	NZ,08	N= 45 HEX=2D	ETA	SYMB071
201 8A88	00B7E		413	MVI	U1,01	N= 46 HEX=2E	** SUPERSCRIPT	SYMB071A
201 8A89	00B7F		414	MVI	U2,01	N= 47 HEX=2F	** SUBSCRIPT	SYMB071B
205 89F9	00AEF		415	MVI	SV,05	N= 48 HEX=30	SUMATION	SYMB071C
20E 89EB	00AE1		416	MVI	SW,14	N= 49 HEX=31	DIVIDE	SYMB071D
206 8684	0077A		417	MVI	SB,06	N= 50 HEX=32	LESS THAN OR EQUAL	SYMB071E
206 8688	0077E		418	MVI	SX,06	N= 51 HEX=33	GREATER OR EQUAL	SYMB071F
204 89FE	00AF4		419	MVI	SZ,04	N= 52 HEX=34	DELTA	SYMB071G
204 868E	00784		420	MVI	TO,04	N= 53 HEX=35	LEFT BRACE	SYMB071H
204 869C	00786		421	MVI	SF,04	N= 54 HEX=36	RIGHT BRACE	SYMB071I
205 8699	0078F		422	MVI	T3,05	N= 55 HEX=37	REVERSE SLASH	
207 8A02	00AF9		423	MVI	T5,07	N= 56 HEX=38	GAMMA	SYMB072A
205 89D2	00AC8		424	MVI	T6,05	N= 57 HEX=39	SQUARE ROOT	SYMB072B
20A 8A09	00AFF		425	MVI	T7,10	N= 58 HEX=3A		SYMB072C
20D 8A09	00AFF		426	MVI	T7,13	N= 59 HEX=3B		SYMB072D
205 8A16	00B0C		427	MVI	T8,05	N= 60 HEX=3C	LEFT ARROW	SYMB072E
20F 8A23	00B19		428	MVI	NI,15	N= 61 HEX=3D	TIMES	
205 89DA	C0AD0		429	MVI	SI,05	N= 62 HEX=3E	UP ARROW	SYMB072G
205 89D7	00ACD		430	MVI	SH,05	N= 63 HEX=3F	DN ARROW	SYMB072H
201 8A87	00B7D		431	MVI	U0,01	N= 64 HEX=40	BLANK	SYMB072I
211 86B7	007AD		432	MVI	LA,17			
213 86C8	007BE		433	MVI	LB,19			
20D 86DB	007D1		434	MVI	LC,13			
20F 86E8	007DE		435	MVI	LD,15			
20D 86F7	007ED		436	MVI	LE,13			
20D 8704	007FA		437	MVI	LF,13			
20F 8711	008C7		438	MVI	LG,15			
20F 8720	C0816		439	MVI	LH,15			
207 872F	00825		440	MVI	LI,7			

```
3JECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
208 8736      0082C          441          MVI   TE,11      N= 74 HEX=4A   CENT                    SYMB073I
205 8757      0084D          442          MVI   EX,05      N= 75 HEX=4B   PERIOD
203 8684      0077A          443          MVI   SB,03      N= 76 HEX=4C   LESS                    SYMB074A
204 8777      0086D          444          MVI   T9,04      N= 77 HEX=4D   LEFT PAREN               SYMB074B
20D 86A3      00799          445          MVI   PL,13      N= 78 HEX=4E   PLUS
202 877B      00371          446          MVI   SQ,02      N= 79 HEX=4F   VERTICAL                SYMB074D
217 877D      00973          447          MVI   SR,23      N= 80 HEX=50   AMPERSAND
208 8794      0088A          448          MVI    LJ,11
212 879F      0C895          449          MVI    LK,18
209 8781      008A7          450          MVI    LL,09
215 878A      008B0          451          MVI    LM,21
212 87CF      008C5          452          MVI    LN,18
20D 87E1      008D7          453          MVI    LO,13
20D 87FE      008E4          454          MVI    LP,13
213 87FB      008F1          455          MVI    LQ,19
211 880E      00904          456          MVI    LR,17
20D 8757      0084D          457          MVI   EX,13      N= 90 HEX=5A   EXCLAMATION
20C 8831      00927          458          MVI   TR,12      N= 91 HEX=5B   DOLLAR SIGN              SYMB075F
213 883D      C0933          459          MVI   TX,19      N= 92 HEX=5C   ASTERISK
204 8850      00946          460          MVI   NA,04      N= 93 HEX=5D   RIGHT PARENTHESIS        SYMB075H
20F 8747      0083D          461          MVI   CM,15      N= 94 HEX=5E   SEMI COLON
203 8854      0094A          462          MVI   NB,03      N= 95 HEX=5F   NOT                     SYMB076
207 8680      007A6          463          MVI   ND,07      N= 96 HEX=60   MINUS
205 869E      00794          464          MVI   T4,05      N= 97 HEX=61   SLASH
211 8859      0094F          465          MVI    LS,17
208 8F6A      C0960          466          MVI    LT,11
209 8875      0096B          467          MVI    LU,09
20C 887E      00974          468          MVI    LV,12
212 888A      C0980          469          MVI    LW,18
215 889C      00992          470          MVI    LX,21
20E 88B1      009A7          471          MVI    LY,14
211 88BF      009B5          472          MVI    LZ,17
211 89BB      00AB1          473          MVI   NO,17      N=106 HEX=6A   INFINITY                SYMB077A
209 8747      0083D          474          MVI   CM,09      N=107 HEX=6B   COMMA
20E 88D0      009C6          475          MVI   NC,14      N=108 HEX=6C   PER CENT                SYMB077C
207 86B0      007A6          476          MVI   ND,07      N=109 HEX=6D   DASH
203 868B      00781          477          MVI   SY,03      N=110 HEX=6E   GREATER THAN            SYMB077E
213 8764      0085A          478          MVI   QU,19      N=111 HEX=6F   QUESTION MARK
219 88E0      009D6          479          MVI    N0,25       N=112 HEX=70  0
208 88F9      C09EF          480          MVI    N1,11       N=113 HEX=71  1
20D 8904      009FA          481          MVI    N2,13                     2
20F 8911      00AC7          482          MVI    N3,15                     3
208 8920      00A16          483          MVI    N4,11                     4
20D 892B      C0A21          484          MVI    N5,13                     5
211 8938      00A2E          485          MVI    N6,17                     6
20D 8949      00A3F          486          MVI    N7,13                     7
215 8956      00A4C          487          MVI    N8,21                     8
20E 896B      00A61          488          MVI    N9,14                     9
208 8751      00847          489          MVI   CO,11      N=122 HEX=7A   COLON
208 8979      00A6F          490          MVI   NE,11      N=123 HEX=7B   POUND                   SYMB078H
210 8984      00A7A          491          MVI   NF,16      N=124 HEX=7C   AT                      SYMB078I
209 8994      00A8A          492          MVI   NG,09      N=125 HEX=7D   SINGLE QUOTE
208 8980      00AA6          493          MVI   EQ,11      N=126 HEX=7E   EQUAL
213 899D      00A93          494          MVI   GN,19      N=127 HEX=7F   DOUBLE QUOTES            SYMB079C
224040404424                 495 SO       DC    X'224040404424'
```

```
BJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

22414030110304l                496 Sl         DC      X'2224140301103041433424'                    SYMB079D
224014124                      497 S2         DC      X'2224014124'                                SYMB079E
2242022024 2                   498 S3         DC      X'222420220242'                              SYMB079F
20440220044                    499 S4         DC      X'220440220044'                              SYMB079G
22402204224                    500 S5         DC      X'222402204224'                              SYMB079H
22024024224                    501 S6         DC      X'222024024224'                              SYMB079I
200440440                      502 S7         DC      X'2200440440'                                SYMB080
24404440C4000                  503 S8         DC      X'22440444C04000'                            SYMB080A
20422442220                    504 S9         DC      X'220422442220'                              SYMB080B
24433130413110 0               505 SA         DC      X'22443313041311001131403133'                SYMB080C
244044000                      506 SC         DC      X'2244044000'                                SYMB080D
22420                          507 SD         DC      X'222420'                                    SYMB080E
226                            508 SG         DC      X'2226'                                      SYMB080F
767F06525F0                    509 SJ         DC      X'2767F06525F0'                              SYMB080G
363F0                          510 SN         DC      X'2363F0'                                    SYMB080H
448462666                      511 SS         DC      X'4448462666'                                SYMB080I
464F06626F03357                512 SM         DC      X'2464F06626F03357'                          SYMB081
82664F0                        513 SB         DC      X'682664F0'                                  SYMB081A
323F0                          514 SX         DC      X'6323F0'                                    SYMB081B
46628                          515 SY         DC      X'246628'                                    SYMB081C
9                              516 T0         DC      X'69'                                        SYMB081D
9                              517 T1         DC      X'29'                                        SYMB081E
262                            518 SF         DC      X'2262'                                      SYMB081F
9292656262262                  519 T2         DC      X'692926562622 62'                           SYMB081G
962723929                      520 T3         DC      X'2962723929' BACK SLASH
232796922                      521 T4         DC      X'2232796922'  SLASH
45455656656 5747               522 PL         DC      X'44545565665657474636354544' PLUS
5556756262435                  523 ND         DC      X'35556756262435' DASH
666684846F04565                524 LA         DC      X'4666684846F045656272793936262 24245'
543636545F04656                525 LB         DC      X'4543636545F04656584846F066692922727666'
338585777792922                526 LC         DC      X'33385857777929227274645333'
3536467584843F0                527 LD         DC      X'43536467584843F0627378692 92262'
345757636387879                528 LE         DC      X'43457576363878792922727343'
245656646487879                529 LF         DC      X'42456566464878792926363242'
348585777792922                530 LG         DC      X'434858577779292272765655656343'
454527276666959                531 LH         DC      X'44545272766669595535392922 4244'
5325259494535                  532 LI         DC      X'35325259494535'
657372624335364                533 TE         DC      X'6657372624335364F04248'                    SYMB082F
536464535F0                    534 TF         DC      X'3536464535F0'                              SYMB082G
442525140506164                535 CM         DC      X'4442525140506164 44F0'  ENTER COMMA AND SEMI COLON
565674745F0                    536 CO         DC      X'4565674745F0' ENTER COLON
442626444F0                    537 EX         DC      X'4442626444F0' LOWER ENTER EXCLAMATION, PERIOD
5656949465655                  538            DC      X'55556949465655' FINISHED
342525343F04454                539 QU         DC      X'4342525343F0445455777929273738686 74544' QUESTION ?
9585362                        540 T9         DC      X'69585362'                                  SYMB082I
12A                            541 SQ         DC      X'212A'                                      SYMB083
4435344F0457579                542 SR         DC      X'44435344F04575793936262 27274646345F04666684846'  &&
424227276666959                543 LJ         DC      X'44242272766569595 34344'
555646272756677                544 LK         DC      X'4555646272756677796967563 63929224245'
3493935252 27273               545 LL         DC      X'43493935252 2727343'
432424434F03646                546 LM         DC      X'3432424434F0364657527279594838291912222736'
444425254F06462                547 LN         DC      X'5444425254F064627279695539292 2323864'
3535666683833F0                548 LO         DC      X'33535666683833F02272792922'
666683836F04575                549 LP         DC      X'3666683836F045757929224245'
344546368383747                550 LQ         DC      X'5344546368383747 4353F0627181727929 2262'
```

```
3JECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

556583836F04555          551 LR       DC    X'365658383 6F045555272766663 29224245'
944F0                    552 ST       DC    X'4944F0'                                          SYMB084A
2435232F0444656          553 SU       DC    X'32435232F04446566768593928'                      SYMB084B
969                      554 TT       DC    X'2969'                                            SYMB084C
942F02353646556          555 TR       DC    X'4942F0235364655636273868'                        SYMB084D
443535463645566          556 TX       DC    X'4443535463645566675657474637364534 3344'  ASTERISK
9383322                  557 NA       DC    X'29383322'                                        SYMB084F
57574                    558 NB       DC    X'257574'                                          SYMB084G
667                      559          DC    X'5667'
424227276363868          560 LS       DC    X'44242272763638686777792925656343 44'
26266565878792 9         561 LT       DC    X'4262665658787929284842'
339292272795953          562 LU       DC    X'33392922727 9595333'
539292432527479          563 LV       DC    X'353929243252747959544335'
429191322324352          564 LW       DC    X'24291913223243526273795953 4449393324'
565627275667779          565 LX       DC    X'45 5627275667779696756463739292736252 24245'
536392926353252          566 LY       DC    X'45363929263532525566695956 45'
47779292737386 8          567 LZ       DC    X'34777929273738686724227274 54533334'
828293938F06922          568 NC       DC    X'3828293938F06922F05363625253'                    SYMB085C
575                      569          DC    X'1575'                    OLD MINUS
634545636F02724          570 NO       DC    X'3634545636F027243353646758382 7F018132262737869 2918'
525226265454929          571 N1       DC    X'3525226265454929283835'
565692928585626          572 N2       DC    X'3565692928535626226263333 5'
545432322626656          573 N3       DC    X'2545432322626656592928484626 25'
454526266565535          574 N4       DC    X'245452626656553539292 4'
555532322626636          575 N5       DC    X'2555532322626636386869292 5'
433535434F03538          576 N6       DC    X'3433535434F0353848475759292 2626535'
542525566692927          577 N7       DC    X'4542525566692927373858564 5'
533535535F03656          578 N8       DC    X'3533535535F03656583836F026 1512727566692926'
656583836FC2545          579 N9       DC    X'3656583836F025 45425262692925'
464545357566626          580 NE       DC    X'2464545357566626363733'                          SYMB086C
657473635445465          581 NF       DC    X'665747363544546567583827243 35364'               SYMB086D
555666949475756          582 NG       DC    X'455566694947575645' SINGLE QUOTE
625354649292737          583 GN       DC    X'362535464929273736F066556576795 76766' DOUBLE QUOTES
534646535F03666          584 EQ       DC    X'3534646535F03666673736' EQUALS
7574637271615 24         585 NO       DC    X'67574637271615 24'                               SYMB086F
445464554647576          586          DC    X'3445464554647576 67'                             SYMB086G
25872268632              587 SE       DC    X'325872268632'                                    SYMB086H
334424A7A                588 T6       DC    X'2334424A7A'                                      SYMB086I
23454                    589 SH       DC    X'423454'                                          SYMB087
249375749                590 SI       DC    X'4249375749'                                      SYMB087A
575565475                591 SK       DC    X'1575565475'                                      SYMB087B
24662                    592 SL       DC    X'224662'                                          SYMB087C
181                      593 SO       DC    X'1181'                                            SYMB087D
A8A                      594 TL       DC    X'1A8A'                                            SYMB087E
565F03334444333          595 SW       DC    X'1565F03334444333F03637474636'                    SYMB087F
121562A7A                596 SV       DC    X'7121562A7A'                                      SYMB087G
2477212                  597 SZ       DC    X'12477212'                                        SYMB087H
8394842485968            598 T5       DC    X'28394842485968'                                  SYMB087I
244345444483858          599 T7       DC    X'42443454444838584849463656'                      SYMB088
536341575                600 T8       DC    X'1536341575'                                      SYMB088A
2213142495A6A79          601 NH       DC    X'12213142495A6A79'                                SYMB088B
454636455666756          602 NI       DC    X'44546364556667564637364534 3344' TIMES
45465665727              603 NJ       DC    X'245465665727'                                    SYMB088D
64266                    604 NK       DC    X'264266'                                          SYMB088E
42535536374F076          605 NL       DC    X'142535536374F0766555372716'                      SYMB088F
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

131424556474A3B             606 NM       DC    X'21314245564 74A3B2B'                                    SYMB088G
151424536474A5B             607 NN       DC    X'61514245364 74A5B6B'                                    SYMB088H
237344353646764             608 NP       DC    X'2237344353646764 73'                                    SYMB088I
33727675753                 609 NQ       DC    X'333727675753'                                           SYMB089
5363443545666F0             610 NT       DC    X'26363443545666F0'                                       SYMB089A
248F0                       611 NR       DC    X'4248F0'                                                 SYMB089B
726243353646657             612 NS       DC    X'372624335364665737F01575'                               SYMB089C
366F026365363               613 NU       DC    X'2366F026365363'                                         SYMB089D
0                           614 U0       DC    X'F0'                                                     SYMB089E
1                           615 U1       DC    X'F1'       ENTER SUPERSCRIPT MODE OR LEAVE SUBSCRIPT.    SYMB089F
2                           616 U2       DC    X'F2'       ENTER SUBSCRIPT MODE OR LEAVE SUPERSCRIPT.    SYMB089G
3                           617 U3       DC    X'F3'       CARRIAGE RETURN.                              SYMB089H
4                           618 U4       DC    X'F4'       BACKSPACE CHARACTER.                          SYMB089I
5                           619 U5       DC    X'F5'       NULL CHARACTER                                SYMB090
724334447445364             620 NV       DC    X'2724334447445364 67'                                    SYMB090A
5374563F02245               621 NW       DC    X'28374563FC2245'                                         SYMB090B
348374656656453             622 NX       DC    X'584837465665645343343546'                               SYMB090C
74736344363F065             623 NY       DC    X'674736344363F06525'                                     SYMB090D
537463446576652             624 NZ       DC    X'2637463446576652'                                       SYMB090E
                            625 STATE    DS    C                                                         SYMB090F
                            626          END                                                             SYMB090G
```

```
BJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    NUMB0000
                           300 NUMBER   START 0          CALL NUMBER(X,Y,HGT,FNUM,THETA,N)   V097
                           301 *
                           302 *    THIS SUBPROGRAM IS THE WATFIV VERSION OF THE CALCOMP PLOT ROUTINE
                           303 *    CALLED NUMBER.  IT CONVERTS A REAL VARIABLE OR CONSTANT TO A FIXED
                           304 *    NUMBER THAT IS PLOTTED BY THE SYMBOL ROUTINE.
                           305 *
                           306           EXTRN SYMBI   OS-TYPE ENTRY POINT FOR SYMBOL ROUTINE
                           307 *
                           308 *
                           309 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                           310 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                           311 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                           312 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                           313 *
                           314 XENTSPEC EQU 246
                           315 XRET     EQU 1124
                           316 XA1      EQU 1592
                           317 *
                           318 *
                           319 *
                           320 *
                           321           WATSL NUMBER,8,SAVE1,(84,84,84,84,84,82),10,
                           322+          USING R111,11
                           323+          CNOP  0,4
)'.8 D00C          0000C   324+          STM   14,11,12(13) 13 CONTAINS CALLING SAVE AREA'S ADDRESS
5B0 C0F6           00DF6   325+          BAL   11,XENTSPEC(0,12) BRANCH TO XENTSPEC  ROUTINE
)0005E4D4C2C5D9            325+R111      DC    H'0',CL6'NUMBER'
)0000188                   327+          DC    A(SAVE) ADDRESS OF A 24 WORD SAVE AREA
                           328+*    THE MODEL ARGUMENT LIST IS CREATED
+000050                    329+          DC    X'84',AL3(LOC11)
+000054                    330+          DC    X'84',AL3(LOC12)
+000058                    331+          DC    X'84',AL3(LOC13)
+00005C                    332+          DC    X'84',AL3(LOC14)
+000060                    333+          DC    X'84',AL3(LOC15)
2000C64                    334+          DC    X'82',AL3(LOC16)
)000000                    335+          DC    X'10',AL3(0)
                           336+*
                           337+*    GENERATING THE OS-TYPE ARGUMENT LIST TO BE USED BY THE SUBPROGRAM
                           338+*
510 8044           0004C   339+A8L1      BAL   1,*+4*7 AN IN-LINE ARGUMENT LIST IS USED.
)000050                    340+          DC    X'00',AL3(LOC11)
)000054                    341+          DC    X'00',AL3(LOC12)
)000058                    342+          DC    X'00',AL3(LOC13)
)00005C                    343+          DC    X'00',AL3(LOC14)
)000060                    344+          DC    X'00',AL3(LOC15)
)000064                    345+          DC    X'80',AL3(LOC16)
                           346+*
                           347+*    GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                           348+*
7F0 8060           00068   349+          BC    15,*+4*(7-0-0)
                           350+LOC11     DS    F
                           351+LOC12     DS    F
                           352+LOC13     DS    F
                           353+LOC14     DS    F
                           354+LOC15     DS    F
```

```
OJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

                            355+LOC16    DS    F
                            356+         DROP  11
                            357+*
                            358+*
680                         359+         BALR  8,0
                            360+         USING *,8
                            361 *                                  * REGISTER USAGE
                            362 *                                  * R1 - N
                            363 *                                  * R2 - K(TEMP N)
                            364 *                                  * R3 - I
                            365 *                                  * R4 - M
                            366 *                                  * R8 BASE
327 1000            00000   367          LM    2,7,0(1)    LOAD LINKAGE                 * F0 - FNUM      NUMB0090
026 8166            001D0   368          STM   2,6,LINK    SAVE LINKAGE TO SYMBOL       * F2 - AFN       NUMB0100
320 8196            00200   369          LD    2,FIX       CLEAR  F2                                     NUMB0110
300 5000            00000   370          LE    0,0(0,5)    PICK UP FNUM                                  NUMB0120
310 7000            00000   371          L     1,0(0,7)    PICK UP N                                     NUMB0130
221                         372          LTR   2,1         K=N                                           NUMB0140
7D0 8030            0009A   373          BC    13,NNEGA    BRANCH IF K IS NOT POSITIVE . . . . . .       NUMB0150
920 811C            00186   374          CH    2,NINE      IS K GREATER THAN 9                    .      NUMB0160
7C0 8028            00092   375          BC    12,MULT        YES          NO                  .      NUMB0170
610 811C            00186   376          LH    1,NINE         N=9           .                  .      NUMB0180
321                         377          LR    2,1            K=9           .                  .      NUMB0190
300 8175            001E8   378 MULT     ME    0,TEN       FNUM=FNUM*10.0  . .          .     .      NUMB0200
620 8028            00092   379          BCT   2,MULT      K=K-1, IF K GREATER 0,REPEAT ABOVE    .      NUMB0210
020                         380 NNEGA    LPER  2,0         AFN=ABS(FNUM)                . . . . . .   NUMB0220
420 8182            001EC   381          AE    2,RND       ROUND AFN=AFN+0.5                             NUMB0230
320 8196            00200   382          AW    2,FIX       ADD FIX CONSTANT                              NUMB0240
020 818E            001F8   383          STD   2,TEMP      SAVE FIXED AFN                                NUMB0250
780 804C            000B6   384          BC    8,FZRO      SKIP LARGE NUMBER TEST IF ZERO                NUMB0260
503 8196 818E 00200 001F8   385          CLC   FIX(4),TEMP TEST FOR LARGE NUMBER                         NUMB0270
770 80F8            00162   386          BC    7,ERRPRT       BRANCH TO ERRPRT IF TOO LARGE              NUMB0280
820 8192            001FC   387 FZRO     L     2,TEMP+4    PICK UP BINARY INTEGER NUMBER                 NUMB0290
222                         388          LTR   2,2         TEST SIGN BIT                                 NUMB0300
740 80F8            00162   389          BC    4,ERRPRT     IF BIT IS 1, BRANCH TO ERRPRT                NUMB0310
E20 818E            001F8   390          CVD   2,TEMP      CONVERT NUMBER TO DECIMAL                     NUMB0320
3F7 8108 818E 00172 001F8   391          UNPK  DEC(16),TEMP AND UNPACK INTO                              NUMB0330
5F0 8117      00181         392          OI    DEC+15,C'0'  DEC TO DEC+15                              NUMB0340
130 8117            00181   393          LA    3,DEC+15    I=15                                          NUMB0350
221                         394          LTR   2,1         K=N                                           NUMB0360
7D0 807A            000E4   395          BC    13,NNEGB    BRANCH IF K IS NOT POSITIVE . . . . . .       NUMB0370
200 3001 3000 00001 00000   396 MVDIG    MVC   1(1,3),0(3) MOVE DEC+I TO DEC+1+I                 .     .NUMB0380
630                         397          BCTR  3,0         I=I+1                                 .     .NUMB0390
620 806E            00006   398          BCT   2,MVDIG     K=K-1,IF K GREATER 0,REPEAT ABOVE     .     .NUMB0400
24B 3001      00001         399 NNEGB    MVI   1(3),C'.'   MOVE '.' INTO DEC+1+I                 . . .NUMB0410
130 8108            00172   400          LA    3,DEC       I=0                                           NUMB0420
5FC 3000      00000         401 ZRTST    CLI   0(3),C'0'   IS DEC+I A '0'                                NUMB0430
770 8092            000FC   402          BC    7,NTZRO        YES          NO                  .      NUMB0440
130 3001            00001   403          LA    3,1(0,3)       I=I+1         .                  .      NUMB0450
7F0 80B2            000EC   404          BC    15,ZRTST       REPEAT TEST   .                  .      NUMB0460
548 3000      00000         405 NTZRO    CLI   0(3),C'.'   IS DEC+I A '.'  . .                  .      NUMB0470
770 809C            00106   406          BC    7,NTPRD        YES          NO                  .      NUMB0480
630                         407          BCTR  3,0            I=I-1         .                  .      NUMB0490
200                         408 NTPRD    LTER  0,0         IS FNUM NEGATIVE  . .                .      NUMB0500
780 80A8            00112   409          BC    11,FNPOS       YES                    NO                NUMB0510
```

```
BJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                     21 SEP 73

630                          410              BCTR  3,0              I=I-1                 .        NUMB0520
2:0 3000        00000        411              MVI   0(3),C'-'         MOVE '-' INTO DEC+I   .        NUMB0530
140 8119              00183  412 FNPOS        LA    4,DEC+17                                 .        NUMB0540
B43                          413              SR    4,3              M=17-I           . . .        NUMB0550
221                          414              LTR   2,1              K=N                              NUMB0560
7?0 80CA              00134  415              BC    13,NNEGC         BRANCH IF K IS NOT POSITIVE . . . . . .  NUMB0570
04? 819E              001F8  416 STCNT        ST    4,TEMP           STORE   M    IN CALL TO SYMBOL      .  NUMB0580
030 8172              001DC  417              ST    3,LINK+12        STORE DEC+I (OR SUBSTITUTE) IN CALL .  NUMB0590
8F0 8186              001F0  418              L     15,SYMAD           TO SYMBOL                         .  NUMB0600
110 8166              00100  419              LA    1,LINK                                               .  NUMB0610
                             420 *
                             421 *    USING AN OS-TYPE CALL TO SYMBOL THROUGH SYMB1
                             422 *
5?F                          423              BALR  14,15            CALL SYMBOL(X,Y,HGT,**,THETA,***)   .  NUMB0620
                             424 *    EXECUTING A WATFIV TYPE RETURN SEQUENCE.
7F0 C4?4              004?4  425              BC 15,XRET(0,12)  BRANCH TO XENT RTN.
A41                          426 NNEGC        AR    4,1              M=M+N                        . . .  NUMB0670
940 811A              00184  427              CH    4,ONE                                                NUMB0680
720 8084              0011E  428              BC    2,STCNT          IF M GREATER THAN 1, BRANCH TO STCNT . .  NUMB0690
740 80DE              00148  429              BC    4,STZRO          IF M LESS THAN 1,BRANCH . . . . .  NUMB0700
200                          430              LTER  0,0              IF M EQUAL 1 AND FNUM IS NOT   .  NUMB0710
730 8084              0011E  431              BC    11,STCNT          NEGATIVE,BRANCH TO STCNT . . . . . . .  NUMB0720
130 8104              0016E  432 STZRO        LA    3,CONST          SET ADDRESS TO  '-0'         .  NUMB0730
140 0002              00002  433              LA    4,2              SET M=2                             NUMB0740
2?0                          434              LTER  0,0              IF FNUM IS NEGATIVE,                NUMB0750
74? 8084              0011E  435              BC    4,STCNT                              BRANCH TO STCNT . .  NUMB0760
130 8105              0016F  436              LA    3,CONST+1        SET ADDRESS TO '0'                  NUMB0770
140 0001              00001  437              LA    4,1              SET M=1                             NUMB0780
7F0 8084              0011E  438              BC    15,STCNT         BRANCH TO STCNT . . . . . . . . . . . .  NUMB0790
130 8106              00170  439 ERRPRT       LA    3,CONST+2        SET ADDRESS TO '**'                 NUMB0800
140 0002              00002  440              LA    4,2              SET M=2                             NUMB0810
7F0 8084              0011E  441              BC    15,STCNT         BRANCH TO STCNT . . . . . . . . . . . .  NUMB0820
0F05C5C                      442 CONST        DC    C'-0**'                                              NUMB0830
                             443 DEC          DS    18C                                                  NUMB0840
001                          444 ONE          DC    H'1'                                                 NUMB0850
009                          445 NINE         DC    H'9'                                                 NUMB0860
                             446 SAVE         DS    18F                                                  NUMB0870
                             447 LINK         DS    5F                                                   NUMB0880
0                            448              DC    X'80'                                                NUMB0890
C01F8                        449              DC    AL3(TEMP)                                            NUMB0900
1A00000                      450 TEN          DC    E'10.0'                                              NUMB0910
0800000                      451 RND          DC    E'0.5'                                               NUMB0920
0000000                      452 SYMAD        DC    A(SYMB1)
                             453 TEMP         DS    D                                                  . NUMB0940
E0000000000000               454 FIX          DC    X'4E00000000000000'                                  NUMB0950
                             455 SAVE1        DS 24F
                             456              END                                                        NUMB0960
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                          1 SCALE    START
                          2          EXTRN SCALE1
                          3 *
                          4 *      THIS ASSEMBLER SUBPROGRAM IS THE INTERFACE BETWEEN A WATFIV
                          5 *   CALLING PROGRAM AND THE ORIGINAL SCALE ROUTINE WHICH IS NOW CALLED
                          6 *   'SCALE1' AND WAS ORIGINALLY WRITTEN IN FORTRAN IV.  A WATFIV
                          7 *   PROGRAM CALLS SCALE WHICH IN TURN CALLS SCALE1.  THE ARGUMENT
                          8 *   LIST FOR SCALE1 IS IDENTICAL TO THE ONE USED BY SCALE EXCEPT IT
                          9 *   HAS AN ADDITIONAL ARGUMENT.
                         10 *
                         11 *   THE CALLING SEQUENCE FOR THIS ROUTINE IS:
                         12 *        CALL SCALE(ARRAY,AXLEN,NPTS,INC)
                         13 *
                         14 *   THE SIZE OF THE ARRAY VECTOR IS TREATED AS BEING OF VARIABLE
                         15 *   DIMENSION BUT NO VARIABLE DIMENSION ELEMENT APPEARS IN THE ARGUMENT
                         16 *   LIST OF SCALE.  SUBSEQUENTLY THIS ASSEMBLER SUBPROGRAM WAS
                         17 *   WRITTEN TO OBTAIN THE LENGTH OF ARRAY AT RUN-TIME AND TO INSERT IT
                         18 *   IN ITS CALLING ARGUMENT LIST FOR SCALE1 AS THE EXTRA ARGUMENT.
                         19 *
                         20 *   THE CALLING SEQUENCE FOR THE SCALE1 ROUTINE IS:
                         21 *        CALL SCALE(ARRAY,AXLEN,NPTS,INC,N)
                         22 *
                         23 *   WHERE N IS THE VARIABLE DIMENSION ELEMENT FOR ARRAY.
                         24 *
                         25 *
                         26 *
                         27 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                         28 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                         29 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                         30 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                         31 *
                         32 XENTSPEC EQU 246
                         33 XRET     EQU 1124
                         34 XA1      EQU 1592
                         35 *
                         36 *
0                        37          BALR 11,0
                         38          USING *,11
0 1000          00000    39          L 2,0(0,1)    LOAD ADDRESS OF STAR ROUTINE FOR THE ACTUAL
                         40 *   ARGUMENT CORRESPONDING TO ARRAY.
0 200C          0000C    41          L 2,12(0,2)   GET LENGTH OF ARRAY IN REGISTER 2.
0 B0C6          000C8    42          ST 2,TPST11   LENGTH OF ARRAY IS STORED IN A STORAGE
                         43 *   LOCATION REFERENCED BY THE STAR ROUTINE FOR ARRAY.
                         44          DROP 11

                         46          USING R111,11
0                        47          CNOP 0,4
C D00C          0000C    48          STM 14,12,12(13)
0 C0F6          000F6    49          BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC RTN.
0F2C3C1D3C540            50 R111     DC H'0',CL6'SCALE'
000C64                   51          DC A(SAVE1)
                         52 *   THE MODEL ARGUMENT LIST
0000D4                   53          DC X'94',AL3(STAR11)   THE STAR RTN. FOR ARRAY
000E8                    54          DC X'84',AL3(LOC11)
000EC                    55          DC X'82',AL3(LOC12)
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

C00F0                     56           DC X'82',AL3(LOC13)
C0000                     57           DC X'10',AL3(0)
                          58 *
                          59 *   SETTING UP A CALLING ARGUMENT LIST TO CALL THE SCALE1 SUBPROGRAM.
                          60 *
0 B044            0005C   61           LA 14,ICI              RETURN ADDRESS
0 B0AC            00004   62           L 15,ADDR              LOAD ADDRESS OF SCALE
0                         63           CNOP 2,4
F                         64           BALR 1,15              MAKING A CALL TO SCALE1.
00004                     65           DC X'94',AL3(STAR11)   USING STAR RTN OF SCALE TO MATCH
                          66 *                                AGAINST THE STAR RTN. OF SCALE1.
000E8                     67           DC X'84',AL3(LOC11)    ADDRESS OF AXLEN
000EC                     68           DC X'82',AL3(LOC12)    ADDRESS OF NPTS
00000                     69           DC X'82',AL3(LOC13)    ADDRESS OF INC
000C8                     70           DC X'92',AL3(TPST11)   TPST11 CONTAINS THE VARIABLE DIMENSION
00000                     71           DC X'10',AL3(0)        ELEMENT FOR ARRAY.
                          72           DROP 11

0                         74 ICI       BALR 11,0              CONTROL RETURNS FROM SCALE1
                          75           USING *,11
                          76 *   A  WATFIV  TYPE  RETURN
0 C464            00464   77           BC 15,XRET(0,12)       BRANCH TO XRET RTN.
                          78           DS 0F
                          79 SAVE1     DS 24F
00000                     80 ADDR      DC A(SCALE1)
C0000                     81 TPST11    DC F'0'
                          82 *
                          83 *   THE STAR ROUTINE FOR ARRAY.
                          84 *
0C1D9D9C1E840             85           DC H'0',CL6'ARRAY'
0 C638            00638   86 STAR11    BAL 15,XA1(0,12)   BRANCH TO XA1
C0C00                     87 LST11     DC AL1(0),AL3(0)  FIXED BY PROLOQUE
00000                     88           DC AL1(2),AL3(0)  LST IS ADDRESS OF ARRAY
C0C00                     89           DC A(0)  FIXED BY PROLOQUE SINCE IT IS A VARIABLE DIMENSION
000C8                     90           DC X'40',AL3(TPST11)  ADDRESS OF VARIABLE DIMENSION ELEMENT
                          91 *
                          92 *  GENERATING STORAGE LOCATIONS FOR CALL-BY-VALUE ARGUMENTS.
                          93 *
                          94 LOC11     DS F
                          95 LOC12     DS F
                          96 LOC13     DS F
                          97           END
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
                           1 AXIS     START 0
                           2          EXTRN AXIS1
                           3 *
                           4 *        THIS ASSEMBLER SUBPROGRAM IS THE INTERFACE BETWEEN A WATFIV
                           5 *   CALLING PROGRAM AND THE ORIGINAL 'AXIS' SUBPROGRAM WRITTEN IN
                           6 *   FORTRAN IV AND IS NOW CALLED 'AXIS1'. A WATFIV PROGRAM CALLS AXIS
                           7 *   WHICH IN TURN CALLS AXIS1.
                           8 *
                           9 *        THE CALLING SEQUENCE OF AXIS IS:
                          10 *    CALL AXIS(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,DELTAV)
                          11 *
                          12 *   WHERE IBCD MAY BE A HOLLERITH STRING, A VECTOR, OR A SIMPLE
                          13 *   VARIABLE OF EITHER TYPE INTEGER*4 OR REAL*4.   THE ADDRESS OF THE
                          14 *   IBCD ARGUMENT IS OBTAINED AND PASSED ON TO THE AXIS1 SUBPROGRAM
                          15 *   BY TREATING THE IBCD ARGUMENT AS A VECTOR.
                          16 *
                          17 *        THE CALLING SEQUENCE FOR THE AXIS1 SUBPROGRAM IS THE SAME AS -
                          18 *   FOR THE AXIS SUBPROGRAM EXCEPT IBCD IS ALWAYS AN INTEGER*4 VECTOR
                          19 *   CONTAINING A SINGLE ELEMENT.
                          20 *
                          21 *
                          22 *
                          23 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                          24 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                          25 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                          26 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                          27 *
                          28 XENTSPEC EQU 246
                          29 XRET    EQU 1124
                          30 XA1     EQU 1592
                          31 *
                          32 *
                          33         BALR 9,0
                          34         USING *,9
                          35 *
                          36 *   DETERMINING WHAT KIND OF ARGUMENT IBCD IS.
                          37 *
9 1008      00008         38         TM 8(1),X'09'  HOLLERITH STRING ?
2 9044            00046   39         BC 1,CHARK     YES
0 1008      00008         40         TM 8(1),X'90'  VECTOR?
0 9066            00068   41         BC 12,NEXTY    NO.  PROCESS A SIMPLE VARIABLE
                          42 *
                          43 *  IBCD  IS A VECTOR ARGUMENT
                          44 *
2 10C8            00C08   45         L 2,8(0,1)     OBTAIN ADDRESS OF IBCD'S STAR ROUTINE.
2 0004            00004   46         L 3,4(2)       OBTAIN ADDRESS OF IBCD
2 200C            0000C   47         L 2,12(0,2)
0 915A            0015C   48         ST 2,TPST      STORING THE ACTUAL VECTOR LENGTH
0 919E            001A0   49         ST 3,STAR12+4  PASS ADDRESS OF IBCD TO THE STAR ROUTINE OF THE
                          50 *                        CALLING SEQUENCE FOR AXIS1
0 9186            00188   51         LA 2,STAR11
0 90A2            000A4   52         ST 2,IBCD      STORING THE ADDRESS OF ITS STAR RTN.
2 1008      00008         53         TM 8(1),X'02'
0 903C            0003E   54         BC 1,INTV      IS INTEGER*4 TYPE
4 90A2      000A4         55         MVI IBCD,X'94' IS REAL*4 TYPE
```

```
E    ADDR1 ADDR2  STMT   SOURCE STATEMENT
        00C88     56           BC 15,ESEQ
     000A4        57 INTV      MVI IBCD,X'92'
        00088     58           BC 15,ESEQ
                  59 *
                  60 *  IBCD IS A HOLLERITH STRING.
                  61 *
        00C08     62 CHARK     L 2,8(0,1)      OBTAIN ADDRESS OF THE ACTUAL DOPE VECTOR
                  63 *  MOVING THE ACTUAL VECTOR LENGTH INTO THE DUMMY DOPE VECTOR.
2000 00160 CC000  64           MVC DOPE(1),0(2)
        00000     65           L 2,0(2)        OBTAIN ADDRESS OF THE HOLLERITH STRING.
        001A0     66           ST 2,STAR12+4   STORE ADDRESS IN A STAR RTN.
        00160     67           LA 2,DOPE
        000A4     68           ST 2,IBCD       STORING THE ADDRESS OF THE DOPE VECTOR
                  69 *   STORING THE TYPE CODE FOR THE DUMMY ARGUMENT OF IBCD
     000A4        70           MVI IBCD,X'09'
        0C098     71           BC 15,ESEQ
                  72 *
                  73 *  IBCD IS A SIMPLE VARIABLE
                  74 *
        0016C     75 NEXTY     LA 2,LOC        STORING THE ADDRESS OF A STORAGE LOCATION
        000A4     76           ST 2,IBCD       FOR THE CALL-BY-VALUE ARGUMENT IBCD.
        001A0     77           ST 2,STAR12+4   STORING ADDRESS OF IBCD IN ITS STAR RTN.
     00008        78           TM 8(1),X'02'
                  79 *   STORING THE TYPE CODE OF IBCD IN THE MODEL ARGUMENT LIST
        0C084     80           BC 1,INTS       IS INTEGER*4 TYPE
     000A4        81           MVI IBCD,X'84'  IS REAL*4 TYPE
        00038     82           BC 15,ESEQ
                  83 *
                  84 *   IBCD IS A SIMPLE VARIABLE OF TYPE INTEGER*4 AND MAYBE THIS IS A
                  85 *   'SPECIAL' CALL TO SYMBOL.
                  86 *
     000A4        87 INTS      MVI IBCD,X'82'
                  88           DROP 9
                  89 *
                  90 *   THE  ENTRY  SEQUENCE  CODE.
                  91 *
                  92           USING REG11,11
                  93           CNOP 0,4
        0000C     94 ESEQ      STM 14,11,12(13)
        000F6     95           BAL 11,XENTSPEC(0,12) BRANCH TO XENTSPEC RTN.
E24040            96 REG11     DC H'0',CL6'AXIS'
                  97           DC A(SAVE1)
                  98 *  THE MODEL ARGUMENT LIST IS CREATED.
                  99           DC X'84',AL3(XPAGE)
                 100           DC X'84',AL3(YPAGE)
                 101 IBCD      DC X'00',AL3(0)      FILLED IN AT RUN-TIME
                 102           DC X'82',AL3(NCHAR)
                 103           DC X'84',AL3(AXLEN)
                 104           DC X'84',AL3(ANGLE)
                 105           DC X'84',AL3(FIRSTV)
                 106           DC X'84',AL3(DELTAV)
                 107           DC X'10',AL3(0)
                 108           DROP 11

                 110           BALR 11,0            CONTROL RETURNS HERE
```

```
CT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                        111          USING *,11
) B0DE     001A0        112          MVI STAR12+4,X'00'  STORING ZEROES IN THE STAR RTN.
                        113 *
                        114 *   SETTING UP A WATFIV TYPE CALL TO AXIS1
                        115 *
) B032          000F4   116          LA 14,ICI
) B0C2          00184   117          L 15,ADDR
                        118          CNOP 2,4
                        119          BALR 1,15
                        120 *   THE WATFIV TYPE CALLING ARGUMNET LIST
)0164                   121          DC X'84',AL3(XPAGE)
)0168                   122          DC X'84',AL3(YPAGE)
)019C                   123          DC X'92',AL3(STAR12)   POINTER TO STAR RTN. FOR IBCD
:0170                   124          DC X'82',AL3(NCHAR)
)0174                   125          DC X'84',AL3(AXLEN)
)0178                   126          DC X'94',AL3(ANGLE)
)017C                   127          DC X'84',AL3(FIRSTV)
:0180                   128          DC X'84',AL3(DELTAV)
:0000                   129          DC X'10',AL3(0)
                        130          DROP 11

)                       132 ICI      BALR 11,0              CONTROL RETURNS HERE FROM AXIS1
                        133          USING *,11
                        134 *   A  WATFIV  TYPE  RETURN
) C464          00464   135          BC 15,XRET(0,12)       BRANCH TO XRET RTN.
                        136 *  DICTIONARY
                        137          DS 0F
                        138 SAVE1    DS 24F
:0000                   139 TPST     DC F'0'
:0000                   140 DOPE     DC F'0'
                        141 XPAGE    DS F
                        142 YPAGE    DS F
                        143 LOC      DS F
                        144 NCHAR    DS F
                        145 AXLEN    DS F
                        146 ANGLE    DS F
                        147 FIRSTV   DS F
                        148 DELTAV   DS F
)0000                   149 ADDR     DC A(AXIS1)
                        150 *
                        151 *   STAR RTN. FOR IBCD FOR THE CALLING ARGUMENT LIST FOR AXIS1.
                        152 *
) C638          00638   153 STAR11   BAL 15,XA1(0,12)       BRANCH TO XA1
)0000                   154          DC AL1(0),AL3(0)
:0000                   155          DC AL1(2),AL3(0)
:0000                   156          DC A(0)
)015C                   157          DC X'40',AL3(TPST)
                        158 *
                        159 *   STAR RTN. FOR IBCD FOR THE ENTRY SEQUENCE CODE
                        160 *
) C638          00638   161 STAR12   BAL 15,XA1(0,12)       BRANCH TO XA1
0000                    162          DC AL1(0),AL3(0)       ADDRESS OF THE ARRAY IS FIXED AT RUN-TIME
0004                    163          DC AL1(2),AL3(4)       LENGTH IS EXPRESSED IN BYTES
0001                    164          DC A(1)
                        165          END
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                           1 AXISI    START 0
                           2          EXTRN AXISI1
                           3 *
                           4 *        THIS ASSEMBLER SUBPROGRAM IS THE INTERFACE BETWEEN A WATFIV
                           5 *   CALLING PROGRAM AND THE ORIGINAL 'AXISI' SUBPROGRAM WRITTEN IN
                           6 *   FORTRAN IV AND IS NOW CALLED 'AXISI1'. A WATFIV PROGRAM CALLS AXISI
                           7 *   WHICH IN TURN CALLS AXISI1.
                           8 *
                           9 *        THE CALLING SEQUENCE OF AXISI IS:
                          10 *    CALL AXISI(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,DELTAV)
                          11 *
                          12 *   WHERE IBCD MAY BE A HOLLERITH STRING, A VECTOR, OR A SIMPLE
                          13 *   VARIABLE OF EITHER TYPE INTEGER*4 OR REAL*4.   THE ADDRESS OF THE
                          14 *   IBCD ARGUMENT IS OBTAINED AND PASSED ON TO THE AXISI1 SUBPROGRAM
                          15 *   BY TREATING THE IBCD ARGUMENT AS A VECTOR.
                          16 *
                          17 *        THE CALLING SEQUENCE FOR THE AXISI1 SUBPROGRAM IS THE SAME AS
                          18 *   FOR THE AXISI SUBPROGRAM EXCEPT IBCD IS ALWAYS AN INTEGER*4 VECTOR
                          19 *   CONTAINING A SINGLE ELEMENT.
                          20 *
                          21 *
                          22 *
                          23 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                          24 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                          25 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                          26 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                          27 *
                          28 XENTSPEC EQU 246
                          29 XRET    EQU 1124
                          30 XA1     EQU 1592
                          31 *
                          32 *
0                         33          BALR 9,0
                          34          USING *,9
                          35 *
                          36 *   DETERMINING WHAT KIND OF ARGUMENT IBCD IS.
                          37 *
9 1008      00008         38          TM 8(1),X'09'  HOLLERITH STRING ?
0 9044            00046   39          BC 1,CHARK     YES
0 1008      00008         40          TM 8(1),X'90'  VECTOR?
0 9066            00068   41          BC 12,NEXTY    NO.  PROCESS A SIMPLE VARIABLE
                          42 *
                          43 *  IBCD  IS A VECTOR ARGUMENT
                          44 *
0 1008            00008   45          L 2,8(0,1)     OBTAIN ADDRESS OF IBCD'S STAR ROUTINE.
2 0004            00004   46          L 3,4(2)       OBTAIN ADDRESS OF IBCD
C 200C            0000C   47          L 2,12(0,2)
0 915A            0015C   48          ST 2,TPST      STORING THE ACTUAL VECTOR LENGTH
0 919E            001A0   49          ST 3,STAR12+4  PASS ADDRESS OF IBCD TO THE STAR ROUTINE OF THE
                          50 *                         CALLING SEQUENCE FOR AXISI1
0 9186            00188   51          LA 2,STAR11
0 90A2            000A4   52          ST 2,IBCD      STORING THE ADDRESS OF ITS STAR RTN.
2 1008      00008         53          TM 8(1),X'02'
0 903C            0003E   54          BC 1,INTV      IS INTEGER*4 TYPE
4 90A2      000A4         55          MVI IBCD,X'94' IS REAL*4 TYPE
```

```
E    ADDR1 ADDR2  STMT   SOURCE STATEMENT
            00088    56           BC 15,ESEQ
      000A4          57 INTV      MVI IBCD,X'92'
            00088    58           BC 15,ESEQ
                     59 *
                     60 *  IBCD IS A HOLLERITH STRING.
                     61 *
            00008    62 CHARK     L 2,8(0,1)      OBTAIN ADDRESS OF THE ACTUAL DOPE VECTOR
                     63 *  MOVING THE ACTUAL VECTOR LENGTH INTO THE DUMMY DOPE VECTOR.
2000 00160  00C00    64           MVC DOPE(1),0(2)
            00000    65           L 2,0(2)        OBTAIN ADDRESS OF THE HOLLERITH STRING.
            001A0    66           ST 2,STAR12+4   STORE ADDRESS IN A STAR RTN.
            00160    67           LA 2,DOPE
            000A4    68           ST 2,IBCD       STORING THE ADDRESS OF THE DOPE VECTOR
                     69 *    STORING THE TYPE CODE FOR THE DUMMY ARGUMENT OF IBCD
      000A4          70           MVI IBCD,X'09'
            00088    71           BC 15,ESEQ
                     72 *
                     73 *  IBCD IS A SIMPLE VARIABLE
                     74 *
            0016C    75 NEXTY     LA 2,LOC        STORING THE ADDRESS OF A STORAGE LOCATION
            000A4    76           ST 2,IBCD       FOR THE CALL-BY-VALUE ARGUMENT IBCD.
            001A0    77           ST 2,STAR12+4   STORING ADDRESS OF IBCD IN ITS STAR RTN.
      000C8          78           TM 8(1),X'02'
                     79 *    STORING THE TYPE CODE OF IBCD IN THE MODEL ARGUMENT LIST
            00084    80           BC 1,INTS       IS INTEGER*4 TYPE
      0C0A4          81           MVI IBCD,X'84' IS REAL*4 TYPE
            00088    82           BC 15,ESEQ
                     83 *
                     84 *    IBCD IS A SIMPLE VARIABLE OF TYPE INTEGER*4 AND MAYBE THIS IS A
                     85 *    'SPECIAL' CALL TO SYMBOL.
                     86 *
      000A4          87 INTS      MVI IBCD,X'82'
                     88           DROP 9
                     89 *
                     90 *    THE  ENTRY  SEQUENCE  CODE.
                     91 *
                     92           USING REG11,11
                     93           CNOP 0,4
            0000C    94 ESEQ      STM 14,11,12(13)
            000F6    95           BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC RTN.
2C940                96 REG11     DC H'0',CL6'AXISI'
                     97           DC A(SAVE1)
                     98 *  THE MODEL ARGUMENT LIST IS CREATED.
                     99           DC X'84',AL3(XPAGE)
                    100           DC X'84',AL3(YPAGE)
                    101 IBCD      DC X'00',AL3(0)     FILLED IN AT RUN-TIME
                    102           DC X'82',AL3(NCHAR)
                    103           DC X'84',AL3(AXLEN)
                    104           DC X'84',AL3(ANGLE)
                    105           DC X'84',AL3(FIRSTV)
                    106           DC X'84',AL3(DELTAV)
                    107           DC X'10',AL3(0)
                    108           DROP 11

                    110           BALR 11,0           CONTROL RETURNS HERE
```

```
ECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT
                         111           USING *,11
) B0DE      001A0        112           MVI STAR12+4,X'00'  STORING ZEROES IN THE STAR RTN.
                         113 *
                         114 *    SETTING UP A WATFIV TYPE CALL TO AXISI1
                         115 *
) B032           000F4   116           LA 14,ICI
) B0C2           00184   117           L 15,ADDR
                         118           CNOP 2,4
                         119           BALR 1,15
                         120 *    THE WATFIV TYPE CALLING ARGUMENT LIST
00164                    121           DC X'84',AL3(XPAGE)
00168                    122           DC X'84',AL3(YPAGE)
0019C                    123           DC X'92',AL3(STAR12)   POINTER TO STAR RTN. FOR IBCD
00170                    124           DC X'92',AL3(NCHAR)
00174                    125           DC X'84',AL3(AXLEN)
00178                    126           DC X'84',AL3(ANGLE)
0017C                    127           DC X'84',AL3(FIRSTV)
00180                    128           DC X'84',AL3(DELTAV)
00000                    129           DC X'10',AL3(0)
                         130           DROP 11

)                        132 ICI       BALR 11,0              CONTROL RETURNS HERE FROM AXISI1
                         133           USING *,11
                         134 *    A  WATFIV  TYPE  RETURN
) C464           00464   135           BC 15,1124(0,12)       BRANCH TO XRET RTN.
                         136 *  DICTIONARY
                         137           DS 0F
                         138 SAVE1     DS 24F
00000                    139 TPST      DC F'0'
00000                    140 DOPE      DC F'0'
                         141 XPAGE     DS F
                         142 YPAGE     DS F
                         143 LOC       DS F
                         144 NCHAR     DS F
                         145 AXLEN     DS F
                         146 ANGLE     DS F
                         147 FIRSTV    DS F
                         148 DELTAV    DS F
00000                    149 ADDR      DC A(AXISI1)
                         150 *
                         151 *    STAR RTN. FOR IBCD FOR THE CALLING ARGUMENT LIST FOR AXISI1.
                         152 *
0 C638           00638   153 STAR11    BAL 15,XA1(0,12)       BRANCH TO XA1
00000                    154           DC AL1(0),AL3(0)
00000                    155           DC AL1(2),AL3(0)
00000                    156           DC A(0)
0015C                    157           DC X'40',AL3(TPST)
                         158 *
                         159 *    STAR RTN. FOR IBCD FOR THE ENTRY SEQUENCE CODE
                         160 *
0 C638           00638   161 STAR12    BAL 15,XA1(0,12)       BRANCH TO XA1
00000                    162           DC AL1(0),AL3(0)     ADDRESS OF THE ARRAY IS FIXED AT RUN-TIME
00004                    163           DC AL1(2),AL3(4)     LENGTH IS EXPRESSED IN BYTES
00001                    164           DC A(1)
                         165           END
```

```
OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                              1 LINE     START
                              2          EXTRN LINE1
                              3 *
                              4 *        THIS ASSEMBLER SUBPROGRAM IS THE INTERFACE BETWEEN A WATFIV
                              5 *   CALLING PROGRAM AND THE ORIGINAL 'LINE' SUBPROGRAM WRITTEN IN
                              6 *   FORTRAN IV AND IS NOW CALLED 'LINE1'.  A WATFIV PROGRAM CALLS
                              7 *   AXIS WHICH IN TURN CALLS AXIS1.
                              8 *
                              9 *        THE CALLING SEQUENCE OF LINE IS:
                             10 *     CALL LINE(XARRAY,YARRAY,NN,INC,LINTYP,INTEQ)
                             11 *
                             12 *        THE ARGUMENT LIST OF THE LINE1 SUBPROGRAM CONTAINS TWO
                             13 *   EXTRA ARGUMENTS.  SINCE THE ARGUMENTS XARRAY AND YARRAY ARE
                             14 *   DECLARED AS ARRAYS IN AXIS1 WITH VARIABLE DIMENSIONING, THE EXTRA
                             15 *   ARGUMENTS ARE USED TO PASS DOWN THE VARIABLE DIMENSION ELEMENT
                             16 *   FOR EACH ARRAY.
                             17 *
                             18 *        THE CALLING SEQUENCE FOR LINE1 IS:
                             19 *    CALL LINE1(XARRAY,YARRAY,NN,INC,LINTYP,INTEQ,DIM1,DIM2)
                             20 *
                             21 *   WHERE DIM1 IS THE VARIABLE DIMENSION ELEMENT FOR XARRAY AND
                             22 *   DIM2 IS THE VARIABLE DIMENSION ELEMENT OF YARRAY.
                             23 *
                             24 *
                             25 *
                             26 *  THESE INSTRUCTIONS REFER TO DISPLACEMENTS IN THE STARTA ROUTINE.
                             27 *  THEY ARE TO BE UPDATED IF THE STARTA ROUTINE IS MODIFIED CAUSING
                             28 *  THESE DISPLACEMENTS TO BE CHANGED OR IF THESE ROUTINES ARE MADE
                             29 *  INTO ENTRY POINTS AT SOME FUTURE DATE.
                             30 *
                             31 XENTSPEC EQU 246
                             32 XRET    EQU 1124
                             33 XA1     EQU 1592
                             34 *
                             35 *
05B0                         36          BALR 11,0
                             37          USING *,11
5820 1000          00000     38          L 2,0(0,1)          LOAD ADDRESS OF ACTUAL STAR RTN FOR XARRAY
5820 200C          0000C     39          L 2,12(0,2)         OBTAIN LENGTH OF XARRAY
                             40 *
                             41 *   LENGTH OF XARRAY IS STORED IN A STORAGE LOCATION REFERENCED BY THE
                             42 *   STAR ROUTINE FOR THE DUMMY ARGUMENT CORRESPONDING TO XARRAY.
                             43 *
5020 B0E6          000E8     44          ST 2,TPST11           STORING THE VECTOR LENGTH
5820 1004          00004     45          L 2,4(0,1)          LOAD ADDRESS OF ACTUAL STAR RTN FOR YARRAY
5820 200C          0000C     46          L 2,12(0,2)         OBTAIN LENGTH OF YARRAY
                             47 *
                             48 *   LENGTH OF YARRAY IS STORED IN A STORAGE LOCATION REFERENCED BY THE
                             49 *   STAR ROUTINE FOR THE DUMMY ARGUMENT CORRESPONDING TO YARRAY.
                             50 *
5020 B0EA          000EC     51          ST 2,TPST12           STORING THE VECTOR LENGTH
                             52          DROP 11

                             54 *   THE  ENTRY  SEQUENCE  CODE.
                             55          USING R111,11
```

```
OBJECT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT

0700                          56          CNOP 0,4
90EC D00C             0000C   57          STM 14,12,12(13)
4580 C0F6             000F6   58          BAL 11,XENTSPEC(0,12)  BRANCH TO XENTSPEC RTN.
0000D3C9D5C54040              59 R111     DC H'0',CL6'LINE'
00000C84                      60          DC A(SAVE1)
                              61 *   THE MODEL ARGUMENT LIST
940000F8                      62          DC X'94',AL3(STAR11)   STAR ROUTINE FOR XARRAY
94000114                      63          DC X'94',AL3(STAR12)   STAR ROUTINE FOR YARRAY
82000128                      64          DC X'82',AL3(LOC11)    NN
8200012C                      65          DC X'82',AL3(LOC12)    INC
82000130                      66          DC X'82',AL3(LOC13)    LINTYP
82000134                      67          DC X'82',AL3(LOC14)    INTEQ
10000000                      68          DC X'10',AL3(0)

                              70 *
                              71 *   SETTING UP THE CALLING SEQUENCE FOR A CALL TO LINE1
                              72 *
4130 BC58             0007C   73          LA 14,IC1  LOADING THE RETURN ADDRESS
58F0 B0C0             000E4   74          L 15,ADDR  LOAD ADDRESS OF LINE1
0700                          75          CNOP 2,4
051F                          76          BALR 1,15  MAKING THE CALL TO LINE1
940000F8                      77          DC X'94',AL3(STAR11)  USING THE STAR RTN. OF XARRAY TO MATCH
                              78 *   AGAINST THE STAR RTN. OF THE CORRESPONDING DUMMY ARGUMENT IN LINE1
94000114                      79          DC X'94',AL3(STAR12)
82000128                      80          DC  X'82',AL3(LOC11) ADDRESS OF NN
8200012C                      81          DC  X'82',AL3(LOC12) ADDRESS OF INC
82000130                      82          DC  X'82',AL3(LOC13) ADDRESS OF LINTYP
82000134                      83          DC  X'82',AL3(LOC14) ADDRESS OF INTEQ
820000E8                      84          DC X'82',AL3(TPST11) ADDRESS OF XARRAY'S LENGTH
820000EC                      85          DC X'82',AL3(TPST12) ADDRESS OF YARRAY'S LENGTH
10000000                      86          DC X'10',AL3(0)
                              87          DROP 11

05B0                          89 IC1      BALR 11,0   CONTROL RETURNS HERE FROM LINE1
                              90          USING *,11
                              91 *  A  WATFIV  TYPE  RETURN
47F0 C464             00464   92          BC 15,XRET(0,12)      BRANCH TO XRET RTN.
                              93          DS OF
                              94 SAVE1    DS 24F
00000000                      95 ADDR     DC A(LINE1)
00000000                      96 TPST11   DC F'0'
00000000                      97 TPST12   DC F'0'
                              98 *
                              99 *   THE STAR ROUTINE FOR XARRAY.
                             100 *
0000E7C1D9D9C1E8             101          DC H'0',CL6'XARRAY'
45F0 C638             00638  102 STAR11   BAL 15,XA1(0,12)   BRANCH TO XA1
                             103 *   THE NEXT THREE LOCATIONS ARE FILLED AT RUN-TIME BY THE WATFIV
                             104 *                      PROLOQUE.
00000000                     105 LST11    DC AL1(0),AL3(0)  LST IS ADDRESS OF XARRAY
02000000                     106          DC AL1(2),AL3(0)
00000000                     107          DC A(0)
400000E8                     108          DC X'40',AL3(TPST11)  ADDRESS OF VARIABLE DIMENSION ELEMENT
                             109 *
                             110 *   THE STAR ROUTINE FOR YARRAY.
```

```
OBJECT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT

                             111 *
                             112          DC H'0',CL6'YARRAY'
00000E8C1D9D9C1E8            113 STAR12   BAL 15,XA1(0,12)   BRANCH TO XA1
45F0 C638            00638   114 *   THE NEXT THREE LOCATIONS ARE FILLED AT RUN-TIME BY THE WATFIV
                             115 *                    PROLOGUE.
000C0000                     116 LST12    DC AL1(0),AL3(0)
020C0000                     117          DC AL1(2),AL3(0)
03C00000                     118          DC A(0)
400000EC                     119          DC X'40',AL3(TPST12)  ADDRESS OF VARIABLE DIMENSION ELEMENT.
                             120 *
                             121 *   GENERATING THE STORAGE LOCATIONS FOR THE CALL-BY-VALUE ARGUMENTS.
                             122 LOC11    DS F
                             123 LOC12    DS F
                             124 LOC13    DS F
                             125 LOC14    DS F
                             126          END
```

```
17          SUBROUTINEAXISI1(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,DELTAV)
      C     THIS PROGRAM IS PART OF THE WATFIV IMPLEMENTATION ROUTINES WHICH
      C     PERFORM THE SAME OPERATIONS AS THE CALCOMP PLOT ROUTINE CALLED
      C     AXISI. THIS SUBPROGRAM IS CALLED BY THE ASSEMBLER ROUTINE CALLED
      C     AXISI.
      C.....     MODIFICATION FOR INTEGER AXIS ANNOTATION FOR NON-SCIENTIFIC  AXSI0030
      C.....        HISTOGRAMS ETC.   (ONLY CHANGE IS IN CALL TO 'NUMBER')    AXSI0040
      C.....     XPAGE,YPAGE  COORDINATES OF STARTING POINT OF AXIS, IN INCHESAXSI0050
      C.....                                                                  AXSI0060
      C.....     IBCD         AXIS TITLE.                                     AXSI0070
      C.....     NCHAR        NUMBER OF CHARACTERS IN TITLE. + FOR C.C-W SIDE.AXSI0080
      C.....     AXLEN        FLOATING POINT AXIS LENGTH IN INCHES.           AXSI0090
      C.....     ANGLE        ANGLE OF AXIS FROM THE X-DIRECTION, IN DEGREES. AXSI0100
      C.....     FIRSTV       SCALE VALUE AT THE FIRST TIC MARK.              AXSI0110
      C.....     DELTAV       CHANGE IN SCALE BETWEEN TIC MARKS ONE INCH APARTAXSI0120
18          DIMENSION IBCD(1)                                                 AXSI0130
19          KN=NCHAR                                                          AXSI0140
20          A=1.0                                                             AXSI0150
21          IF (KN) 1,2,2                                                     AXSI0160
22        1 A=-A                                                              AXSI0170
23          KN=-KN                                                            AXSI0180
24        2 EX=0.0                                                          AXSI0190
25          ADX= ABS  (DELTAV)                                                AXSI0200
26          IF (ADX) 3,7,3                                                    AXSI0210
      C.....     ALSO ALLOW LARGER (4 DIGIT) INTEGER NUMBERS                  AXSI0220
27        3 IF (ADX- 9999.0) 6,4,4                                            AXSI0230
28        4 ADX=ADX/10.0                                                      AXSI0240
29          EX=EX+1.0                                                         AXSI0250
30          GO TO 3                                                           AXSI0260
31        5 ADX=ADX*10.0                                                      AXSI0270
32          EX=EX-1.0                                                         AXSI0280
33        6 IF (ADX-0.01) 5,7,7                                               AXSI0290
34        7 XVAL=FIRSTV*10.0**(-EX)                                           AXSI0300
35          ADX= DELTAV*10.0**(-EX)                                           AXSI0310
36          STH=ANGLE*0.0174533                                               AXSI0320
37          CTH=COS(STH)                                                      AXSI0330
38          STH=SIN(STH)                                                      AXSI0340
39          DXB=-0.1                                                          AXSI0350
40          DYB=0.15*A-0.05                                                   AXSI0360
41          XN=XPAGE+DXB*CTH-DYB*STH                                          AXSI0370
42          YN=YPAGE+DYB*CTH+DXB*STH                                          AXSI0380
43          NTIC=AXLEN+1.0                                                    AXSI0390
44          NT=NTIC/2                                                         AXSI0400
45          DO  20  I=1,NTIC                                                  AXSI0410
46            CALL NUMBER(XN,YN,0.105,XVAL,ANGLE,-1)                          AXSI0420
47            XVAL=XVAL+ADX                                                   AXSI0430
48            XN=XN+CTH                                                       AXSI0440
49            YN=YN+STH                                                       AXSI0450
50            IF (NT) 20,11,20                                                AXSI0460
51       11     Z=KN                                                          AXSI0470
52              IF (EX)  12,13,12                                             AXSI0480
53       12     Z=Z+7.0                                                       AXSI0490
54       13     DXB=-.07*Z+AXLEN*0.5                                          AXSI0500
55              DYB=0.325*A-0.075                                             AXSI0510
56              XT=XPAGE+DXB*CTH-DYB*STH                                      AXSI0520
57              YT=YPAGE+DYB*CTH+DXB*STH                                      
58              CALL SYMBOL(XT,YT,0.14,IBCD,ANGLE,KN)                         
59              IF (EX)  14,20,14                                             AXSI0540
60       14       Z=KN+2                                                      AXSI0550
61                XT=XT+Z*CTH*0.14                                            AXSI0560
```

```
62                YT=YT+Z*STH*0.14                                          AXSI0570
63                CALL SYMBOL(XT,YT,0.14,1550938112,ANGLE,3)                AXSI0580
64                XT=XT+(3.0*CTH-0.8*STH)*0.14                              AXSI0590
65                YT=YT+(3.0*STH+0.8*CTH)*0.14                              AXSI0600
66                CALL NUMBER(XT,YT,0.07,EX,ANGLE,-1)                       AXSI0610
67     20     NT=NT-1                                                       AXSI0620
68            CALL PLOT(XPAGE+AXLEN*CTH,YPAGE+AXLEN*STH,3)                  AXSI0630
69            DXB=-0.07*A*STH                                               AXSI0640
70            DYB=+0.07*A*CTH                                               AXSI0650
71            A=NTIC-1                                                      AXSI0660
72            XN=XPAGE+A*CTH                                                AXSI0670
73            YN=YPAGE+A*STH                                                AXSI0680
74            DO  30  I=1,NTIC                                              AXSI0690
75              CALL PLOT(XN,YN,2)                                          AXSI0700
76              CALL PLOT(XN+DXB,YN+DYB,2)                                  AXSI0710
77              CALL PLOT(XN,YN,2)                                          AXSI0720
78              XN=XN-CTH                                                   AXSI0730
79              YN=YN-STH                                                   AXSI0740
80     30     CONTINUE                                                      AXSI0750
81            RETURN                                                        AXSI0760
82            END                                                           AXSI0770
```

```
      SUBROUTINE SCALE1(ARRAY,AXLEN,NN,INC,NT)
C     THIS SUBPROGRAM IS PART OF THE WATFIV IMPLEMENTATION OF THE
C     CALCOMP PLOT ROUTINE CALLED SCALE.  THIS SUBPROGRAM IS CALLED BY
C     THE ASSEMBLER SUBPROGRAM CALLED SCALE.
C.....     ARRAY   NAME OF ARRAY CONTAINING VALUES TO BE SCALED.          0040
C.....     AXLEN   LENGTH IN INCHES OVER WHICH ARRAY IS TO BE SCALED.     0050
C.....     NN      NUMBER OF POINTS TO BE SCALED.                         0060
C.....     INC     INCREMENT OF LOCATION OF SUCCESSIVE POINTS.            0070
C.....     NT      LENGTH OF ARRAY.
      DIMENSION ARRAY(NT),SAVE(7)
C  THE ARRAY MUST BE DIMENSIONED AT LEAST TWO LOCATIONS LARGER THAN
C  THE ACTUAL NUMBER OF DATA VALUES IT CONTAINS.
      K=IABS(INC)                                                         0190
C  TESTING TO SEE IF IT'S SIZE IS LARGER
      IF (NN*K+K+1 .LE. NT) GO TO 4
C  IT ISN'T LARGER
C  PRINT OUT ERROR MESSAGE AND  RETURN.
      PRINT,'***ERROR***  THE VECTOR MUST BE DIMENSIONED LARGER THAN THE
     1 NUMBER OF DATA VALUES USED IN SCALE.'
      PRINT,'          SCALE IS NOT EXECUTED.'
      RETURN
    4 SAVE(1) = 1.0                                                       0120
      SAVE(2)=2.0                                                         0130
      SAVE(3)=4.0                                                         0140
      SAVE(4)=5.0                                                         0150
      SAVE(5)=8.0                                                         0160
      SAVE(6)=10.0                                                        0170
      SAVE(7)=20.                                                         0180
      FAD=0.01                                                            0210
      YO=ARRAY(1)                                                         0220
      YN=YO
      N = NN * K
      DO 40 I=1,N,K                                                       0240
      YS=ARRAY(I)                                                         0250
      IF (YO-YS) 20,20,10                                                 0260
10    YO=YS                                                               0270
      GO TO 40                                                            0280
20    IF (YS-YN) 40,40,30                                                 0290
30    YN=YS                                                               0300
40    CONTINUE                                                            0310
      FIRSTV=YO                                                           0320
      IF (YO) 50,60,60                                                    0330
50    FAD=FAD-1.0                                                         0340
60    DELTAV=(YN-FIRSTV)/AXLEN                                            0350
      IF (DELTAV) 180,130,70                                              0360
70    I=ALOG10(DELTAV)+1000.0                                             0370
      P=10.0**(I-1000)                                                    0380
      DELTAV=DELTAV/P-0.01                                                0390
      DO 80 I=1,6                                                         0400
      IS=I                                                                0410
      IF (SAVE(I)-DELTAV) 80,90,90                                        0420
80    CONTINUE                                                            0430
90    DELTAV=SAVE(IS)*P                                                   0440
      FIRSTV=DELTAV*AINT(YO/DELTAV+FAD)                                   0450
      T=FIRSTV+(AXLEN+0.01)*DELTAV                                        0460
      IF (T-YN) 100,120,120                                               0470
100   FIRSTV=P*AINT(YO/P+FAD)                                             0480
      T=FIRSTV+(AXLEN+.01)*DELTAV                                         0490
      IF (T-YN) 110,120,120
```

```
128    110    IS=IS+1                                                          0500
129           GO TO 90                                                         0510
130    120    FIRSTV=FIRSTV-AINT((AXLEN+(FIRSTV-YN)/DELTAV)/2.0)*DELTAV        0520
131           IF (YO*FIRSTV) 130,130,140                                       0530
132    130    FIRSTV=0.0                                                       0540
133    140    IF (INC) 150,150,160                                             0550
134    150    FIRSTV=FIRSTV+AINT(AXLEN+.5)*DELTAV                              0560
135           DELTAV=-DELTAV                                                   0570
136      160  N=NN*K+1
137           ARRAY(N)=FIRSTV                                                  0590
138           N=N+K
139           ARRAY(N)=DELTAV                                                  0610
140    170    RETURN                                                           0620
141    180    DELTAV=2.0*FIRSTV                                                0630
142           DELTAV=ABS(DELTAV/AXLEN)+1.                                      0640
143           GO TO 70                                                         0650
144           END                                                              0660
```

```
145         SUBROUTINE LINE1(XARRAY,YARRAY,NPTS,INC,LINTYP,INTEQ,IDIM1,IDIM2)
      C     THIS SUBPROGRAM IS PART OF THE WATFIV IMPLEMENTATION OF THE
      C     CALCOMP PLOT ROUTINE CALLED LINE.  AN ASSEMBLER SUBPROGRAM
      C     CALLED LINE EXISTS WHICH CALLS THIS SUBPROGRAM.
      C
      C.....      XARRAY  NAME OF ARRAY CONTAINING ABSCISSA OR X VALUES.          0040
      C.....      YARRAY  NAME OF ARRAY CONTAINING ORDINATE OR Y VALUES.          0050
      C.....      NPTS    NUMBER OF POINTS TO BE PLOTTED.                         0060
      C.....      INC     INCREMENT OF LOCATION OF SUCCESSIVE POINTS.             0070
      C.....      LINTYP  CONTROL TYPE OF LINE--SYMBOLS, LINE, OR COMBINATION.    0080
      C.....      INTEQ   INTEGER EQUIVALENT OF SYMBOL TO BE USED, IF ANY.        0090
      C.....      IDIM1   VARIABLE DIMENSION ELEMENT USED BY XARRAY.
      C.....      IDIM2   VARIABLE DIMENSION ELEMENT USED BY YARRAY.
      C
146         DIMENSION XARRAY(IDIM1),YARRAY(IDIM2)                               0130
147         LMIN=NPTS*INC+1                                                     0140
148         LDX=LMIN+INC                                                        0150
149         NL=LMIN-INC                                                         0160
150         FIRSTX=XARRAY(LMIN)                                                 0170
151         DELTAX=XARRAY(LDX)                                                  0180
152         FIRSTY=YARRAY(LMIN)                                                 0190
153         DELTAY=YARRAY(LDX)                                                  0200
154         CALL WHERE(XN,YN,DF)                                                0210
155         DF=AMAX1(ABS((XARRAY(1)-FIRSTX)/DELTAX-XN),ABS((YARRAY(1)-FIRSTY)/  0220
           &DELTAY-YN))                                                         0230
156         DL=AMAX1(ABS((XARRAY(NL)-FIRSTX)/DELTAX-XN),ABS((YARRAY(NL)-FIRSTY  0240
           &)/DELTAY-YN))                                                       0250
157         IPEN=3                                                              0260
158         ICODE=-1                                                            0270
159         NT=IABS(LINTYP)                                                     0280
160         IF (LINTYP) 20,10,20                                                0290
161   10    NT=1                                                                0300
162   20    IF (DF-DL) 40,40,30                                                 0310
163   30    NF=NL                                                               0320
164         NA=((NPTS-1)/NT)*NT+NT-(NPTS-1)                                     0330
165         KK=-INC                                                             0340
166         GO TO 50                                                            0350
167   40    NF=1                                                                0360
168         NA=NT                                                               0370
169         KK=INC                                                              0380
170   50    IF (LINTYP) 60,70,80                                                0390
171   60    IPENA=3                                                             0400
172         ICODEA=-1                                                           0410
173         LSW=1                                                               0420
174         GO TO 90                                                            0430
175   70    NA=LDX                                                              0440
176   80    IPENA=2                                                             0450
177         ICODEA=-2                                                           0460
178         LSW=0                                                               0470
179   90    DO 150 I=1,NPTS                                                     0480
180         XN=(XARRAY(NF)-FIRSTX)/DELTAX                                       0490
181         YN=(YARRAY(NF)-FIRSTY)/DELTAY                                       0500
182         IF (NA-NT) 100,110,120                                              0510
183   100   IF (LSW) 130,120,130                                                0520
      C  21 CALL SYMBOL (XN,YN,0.08,INTEQ,0.0,ICODE)                            0530
184   110   CALL SYMBOL(XN,YN,0.08,INTEQ,0.0,ICODE)                             0540
185         NA=1                                                                0550
186         GO TO 140                                                           0560
187   120   CALL  PLOT(XN,YN,IPEN)
```

```
188    130    NA=NA+1                                                  0570
189    140    NF=NF+KK                                                 0580
190           ICODE=ICODEA                                             0590
191    150    IPEN=IPENA                                               0600
192           RETURN                                                   0610
193           END                                                      0620
```

```
194           SUBROUTINE AXIS1(XPAGE,YPAGE,IBCD,NCHAR,AXLEN,ANGLE,FIRSTV,DELTAV)
      C       THIS PROGRAM IS PART OF THE WATFIV IMPLEMENTATION ROUTINES WHICH
      C       PERFORM THE SAME OPERATIONS AS THE CALCOMP PLOT ROUTINE CALLED
      C       AXIS.  THIS SUBPROGRAM IS CALLED BY THE ASSEMBLER ROUTINE CALLED
      C       AXIS.
      C.....       XPAGE,YPAGE  COORDINATES OF STARTING POINT OF AXIS, IN INCHES    0030
      C.....       IBCD         AXIS TITLE.                                         0040
      C.....       NCHAR        NUMBER OF CHARACTERS IN TITLE. + FOR C.C-W SIDE.    0050
      C.....       AXLEN        FLOATING POINT AXIS LENGTH IN INCHES.               0060
      C.....       ANGLE        ANGLE OF AXIS FROM THE X-DIRECTION, IN DEGREES.     0070
      C.....       FIRSTV       SCALE VALUE AT THE FIRST TIC MARK.                  0080
      C.....       DELTAV       CHANGE IN SCALE BETWEEN TIC MARKS ONE INCH APART    0090
195           DIMENSION IBCD(1)                                                     0110
196           KN=NCHAR                                                            0120
197           A=1.0                                                               0130
198           IF (KN) 10,20,20                                                    0140
199     10    A=-A                                                                0150
200           KN=-KN                                                              0160
201     20    EX=0.0                                                              0170
202           ADX=ABS(DELTAV)                                                     0180
203           IF (ADX) 30,70,30                                                   0190
204     30    IF (ADX-99.0) 60,40,40                                              0200
205     40    ADX=ADX/10.0                                                        0210
206           EX=EX+1.0                                                           0220
207           GO TO 30                                                            0230
208     50    ADX=ADX*10.0                                                        0240
209           EX=EX-1.0                                                           0250
210     60    IF (ADX-0.01) 50,70,70                                              0260
211     70    XVAL=FIRSTV*10.0**(-EX)                                             0270
212           ADX=DELTAV*10.0**(-EX)                                              0280
213           STH=ANGLE*0.0174533                                                 0290
214           CTH=COS(STH)                                                        0300
215           STH=SIN(STH)                                                        0310
216           DXB=-0.1                                                            0320
217           DYB=0.15*A-0.05                                                     0330
218           XN=XPAGE+DXB*CTH-DYB*STH                                            0340
219           YN=YPAGE+DYB*CTH+DXB*STH                                            0350
220           NTIC=AXLEN+1.0                                                      0360
221           NT=NTIC/2                                                           0370
222           DO 120 I=1,NTIC                                                     0380
223           CALL NUMBER(XN,YN,0.105,XVAL,ANGLE,2)                               0390
224           XVAL=XVAL+ADX                                                       0400
225           XN=XN+CTH                                                           0410
226           YN=YN+STH                                                           0420
227           IF (NT) 120,80,120                                                  0430
228     80    Z=KN                                                                0440
229           IF (EX) 90,100,90                                                   0450
230     90    Z=Z+7.0                                                             0460
231     100   DXB=-.07*Z+AXLEN*0.5                                                0470
232           DYB=0.325*A-0.075                                                   0480
233           XT=XPAGE+DXB*CTH-DYB*STH                                            0490
234           YT=YPAGE+DYB*CTH+DXB*STH
235           ITEMP = IBCD(1)
236           CALL SYMBOL(XT,YT,0.14,IBCD,ANGLE,KN)                               0520
237           IF (EX) 110,120,110                                                 0530
238     110   Z=KN+2                                                              0540
239           XT=XT+Z*CTH*0.14                                                    0550
240           YT=YT+Z*STH*0.14                                                    0570
241           CALL SYMBOL(XT,YT,0.14,1550938112,ANGLE,3)
```

```
      XT=XT+(3.0*CTH-0.8*STH)*0.14                                      0580
      YT=YT+(3.0*STH+0.8*CTH)*0.14                                      0590
      CALL NUMBER(XT,YT,0.07,EX,ANGLE,-1)                               0600
      NT=NT-1                                                           0610
120   CALL  PLOT(XPAGE+AXLEN*CTH,YPAGE+AXLEN*STH,3)                     0620
      DXB=-0.07*A*STH                                                   0630
      DYB=+0.07*A*CTH                                                   0640
      A=NTIC-1                                                          0650
      XN=XPAGE+A*CTH                                                    0660
      YN=YPAGE+A*STH                                                    0670
      DO 130 I=1,NTIC                                                   0680
      CALL  PLOT(XN,YN,2)                                               0690
      CALL  PLOT(XN+DXB,YN+DYB,2)                                       0700
      CALL  PLOT(XN,YN,2)                                               0710
      XN=XN-CTH                                                         0720
      YN=YN-STH                                                         0730
130   CONTINUE                                                          0740
      RETURN                                                            0750
      END                                                               0760
```

```
261          SUBROUTINE ERROR1(I)
262          GO TO (10,20,30),I
263       10   PRINT,'***ERROR***  ABS(SIN OR COS FUNCTION ARGUMENT) >= PI * 2
            1** 18.'
264          RETURN
265       20 PRINT,'***ERROR***  ATTEMPT TO MOVE LEFT FURTHER THAN PLOT REGION
            1ALLOWS.'
266          PRINT,'               MOVE PLOT ORIGIN(S) AN APPROPRIATE NUMBER OF I
            1NCHES TO THE RIGHT.'
267          RETURN
268       30 PRINT,'***WARNING***  CLOSING X COORDINATE POSITION SHOULD BE BEYO
            1ND RIGHTMOST X POINT PLOTTED.'
269          RETURN
270          END

      $ENTRY

CORE USAGE       OBJECT CODE=   22760 BYTES,ARRAY AREA=   16236 BYTES,TOTAL AREA AVAILABLE=   8(

DIAGNOSTICS        NUMBER OF ERRORS=       0, NUMBER OF WARNINGS=       0, NUMBER OF EXTENSIONS=

COMPILE TIME=     2.21 SEC,EXECUTION TIME=     0.78 SEC,DATE=  73/263,TIME=21:47:42
      OSTOP
```

```
:CT CODE    ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    00020018
                          1 IHCSSCN       CSECT
                          2 *
                          3 *        THIS ROUTINE WAS MODIFIED SO THAT THE SIN AND COS
                          4 *    FUNCTIONS MAY BE CALLED (USING THE NAMES SIN1 AND COS1) BY THE
                          5 *    MODIFIED CALCOMP PLOT ROUTINES WHICH RUN UNDER WATFIV.          00030018
                          6 *                                                                    00040018
                          7 *      SINE-COSINE FUNCTIONS (SHORT)                                 00050018
                          8 *              1. DIVIDE MAGNITUDE OF ARG BY PI/4 TO FIND OCTANT     00060018
                          9 *                   AND FRACTION.                                    00070018
                         10 *              2. IF COSINE, CRANK OCTANT NUMBER BY 2.               00080018
                         11 *              3. IF SINE, CRANK OCTANT NUMBER BY 0(4) FOR +ARG(-ARG). 00090018
                         12 *              4. COMPUTE SINE OR COSINE OF FRACTION*PI/4 DEPENDING  00100018
                         13 *                   ON THE OCTANT.                                   00110018
                         14 *              5. IF OCTANT NUMBER IS FOR LOWER PLANE, MAKE SIGN -.

                         16        ENTRY   COS1            OS-TYPE ENTRY POINT
                         17        ENTRY   SIN1            OS-TYPE ENTRY POINT
                         18          EXTRN ERROR1          ERROR MESSAGE RTN.

                         20 GRA    EQU     1               ARGUMENT POINTER                     00310018
                         21 GRS    EQU     13              SAVE AREA POINTER                    00320018
                         22 GRR    EQU     14              RETURN REGISTER                      00330018
                         23 GRL    EQU     15              LINK REGISTER                        00340018
                         24 GR0    EQU     0               SCRATCH REGISTERS                    00350018
                         25 GR1    EQU     1                                                    00360018
                         26 GR2    EQU     14                                                   00370018
                         27 *        GENERAL REGISTERS 2 AND 3 ALSO USED                        00430016
                         28 FR0    EQU     0               ANSWER REGISTER                      00440000
                         29 FR2    EQU     2               SCRATCH REGISTERS                    00460000
                         30 FR4    EQU     4                                                    00480000
                         31 ON       EQU   X'FF'                                               00486014
                         32 OFF      EQU   X'00'                                               00492014

                         34        USING   *,GRL                                               00510018
0 F008          00008    35 COS1   BC      15,SCOS         COSINE ENTRY
                         36        DC      AL1(3)                                               00530018
                         37        DC      CL3'COS'                                             00540018
5C2
F D00C          0000C    38 SCOS   STM     GRR,GRL,12(GRS) SAVE REGISTERS                       00550018
2 F0F3   000F3          39        MVI     CRANK+3,X'02'     FOR COSINE, OCTANT CRANK IS 2      00560018
1 0000          00000    40        L       GR2,0(GRA)          COS(X) = SIN(PI/2+X)           00570018
0 F038          00038    41        BAL     GRL,MERGE             ADJUST BASE REGISTER AND MERGE 00580018

                         43        USING   *,GRL                                               00600018
C F008          00020    44 SIN1   BC      15,SSIN         SINE ENTRY
                         45        DC      AL1(3)                                               00620018
                         46        DC      CL3'SIN'                                             00630018
9D5
F D00C          0000C    47 SSIN   STM     GRR,GRL,12(GRS) SAVE REGISTERS                       00640018
1 0000          00000    48        L       GR2,0(GRA)                OCTANT CRANK IS 4 IF -ARG  00650018
0 F00B   000F3          49 SIGN   MVI     CRANK+3,X'00'     FOR SINE, OCTANT CRANK IS 0 IF +ARG 00660018
0 F000   00000          50        TM      0(GR2),X'80'        SIN(-X) = SIN(PI+X)              00670018
0 F020          00038    51        BC      8,*+8                                                00680018
4 F00B   000F3          52        MVI     CRANK+3,X'04'                                        00690018

0 F00D          000E8    54 MERGE  LD      FR4,ONE         LOAD FR4 DOUBLE WITH ONE             00710018
0 F00B          000F0    55        LD      FR2,CRANK       CLEAR L.O. FR2 AND LOAD WITH CRANK   00720018
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT                                                    00730018
                          56         LDR     FR0,FR2          CLEAR L.O. FR0                         00740018
 0000               00000 57         LE      FR0,0(GR2)       OBTAIN ARGUMENT                        00750018
                          58         LPER    FR0,FR0          CONSIDER ARGUMENT TO BE POSITIVE       00760018
 F0EC               00104 59         CE      FRC,MAX                                                 00770018
 F09E               00DB6 60         BC      10,ERROR         ERROR IF /X/ GRT THAN OR = PI*2**18    00790018
 F0E0               00DF8 61         MD      FR0,F3VPI        MULTIPLY BY 4/PI (LONG FORM)           00810018
                          62         CER     FR0,FR4                                                 00830018
 F048               00060 63         BC      4,SMALL          IF PRODUCT LESS THAN 1, JUMP           00850018
                          64         AWR     FR0,FR2          GIVE PROD CHAR OF 46, UNNORM, ADD CRANK 00870018
                          65         LER     FR2,FR0          INTEGER PART OF PROD TO FR2, UNNORM    00890018
                          66         SDR     FR0,FR2          FRACTION PART OF PROD TO FR0, NORM      00930018
 F0B4               000CC 68 SMALL   STE     FR2,OCTNT        SAVE OCTANT. LAST 3 BITS ARE MOD OCTANT 00950018
 F0B7      000CF    00070 69         TM      OCTNT+3,X'01'    IF ODD OCTANT, TAKE COMPLEMENT OF       00970018
 F058               00070 70         BC      8,EVEN             FRACTION TO OBTAIN THE MODIFIED       00990018
                          71         SDR     FR0,FR4              FRACTION R                          01010018
                          72         LPER    FR0,FR0                                                  01050018
 0004               00004 74 EVEN    LA      GR1,4            GR1 = 4 FOR COSINE POLYNOMIAL          01070018
 F0B7      000CF    00080 75         TM      OCTNT+3,X'03'      THIS IS FOR OCTANT 2, 3, 6, OR 7     01090018
 F068               00080 76         BC      4,*+8            GR1 = 0 FOR SINE POLYNOMIAL            01110018
                          77         SR      GR1,GR1            THIS IS FOR OCTANT 1, 4, 5, OR 8     01130018
                          78         LER     FR4,FR0          LOAD FR4 WITH R FOR MULTIPLICATION     01150018
 F0E8               00100 79         CE      FR0,UNFLO                                               01170018
 F072               0CC8A 80         BC      2,*+6            IF R**2 LST 16**-3, SET TO 0           01190018
                          81         SER     FR0,FR0            THIS AVOIDS IRRELEVANT UNDERFLOW     01210018
                          82         MER     FR0,FR0          COMPUTE SINE OR COSINE OF MODIFIED     01230018
                          83         LER     FR2,FR0            FRACTION USING PROPER CHEBYSHEV      01250018
 F0C0               000D8 84         ME      FR0,S3(GR1)          INTERPOLATION POLYNOMIAL          01270018
 F0C4               000DC 85         AE      FR0,S2(GR1)                                              01290018
                          86         MER     FR0,FR2                                                  01310018
 F0C8               000E0 87         AE      FR0,S1(GR1)                                              01330018
                          88         MER     FR0,FR2                                                  01350018
 F0CC               000E4 89         AE      FR0,S0(GR1)      SIN(R)/R OR COS(R) READY               01370018
                          90         MER     FR0,FR4          IF SINE POLYNOMIAL, MULTIPLY BY R      01390018
 F0B7      000CF          91         TM      OCTNT+3,X'04'                                           01410018
 F096               000AE 92         BC      8,*+6            IF MODIFIED OCTANT IS IN               01430018
                          93         LNER    FR0,FR0            LOWER PLANE, SIGN IS NEGATIVE        01643016
                          95 EXIT    EQU     *                                                       01651018
 000C               0000C 96         L       GPR,12(GRS)                                             01680000
                          97 *       MVI     12(GRS),X'FF'    RETURN
                          98         BCR     15,GRR
                         100 *
                         101 *
                         102           CNOP 2,4
                         103 *
                         104 *  CALLING SEQUENCE CODE FOR ERROR1
                         105 *
 F0B0               000C8 106 ERROR     LA 14,ICI          LOADING THE RETURN ADDRESS
 F0B8               000D0 107           L 15,ADERR         ADDRESS OF THE CALLED RTN ERROR1.
                         108           BALR 1,15
                         109 *  THE CALLING ARGUMENT LIST
 000D4                   110           DC X'82',AL3(ITEMP)
```

```
CT CODE     ADDR1 ADDR2  STMT   SOURCE STATEMENT
)0000                     111            DC X'10',AL3(0)
                          112 ICI        EQU *
) F096                    113            BC 15,EXIT                                                    01801018
                          115 OCTNT  DS       F                                                        01802018
                          116        DS       0D
:0000                     117 ADERR     DC A(ERROR1)
)0001                     118 ITEMP     DC F'1'
                          119 S3     DC       X'BD25B368'    -0.00003595   SIN C3                    01804018
                          120 S2     DC       X'3EA32F62'     0.00249001   SIN C2                    01806018
                          121 S1     DC       X'C014ABBC'    -0.08074543   SIN C1                    01808018
                          122 S0     DC       X'40C90FDB'     0.78539816   SIN C0                    01810018
                          123 C0     DC       X'41100000'     1.00000000   COS C0                    01811018
                          124        DC       X'CC000000'                                             01812018
                          125 ONE    EQU      C0                                                      01813018
                          126 CRANK  DC       X'4600000000000000'                                     01814018
                          127 FOVPI  DC       X'41145F306DC9C883'                                     01815018
                          128 UNFLO  DC       X'3E100000'                                             01816018
                          129 MAX    DC       X'45C90FDA'       PI*2**18                              01817018
                          130        END                                                             02140000
```

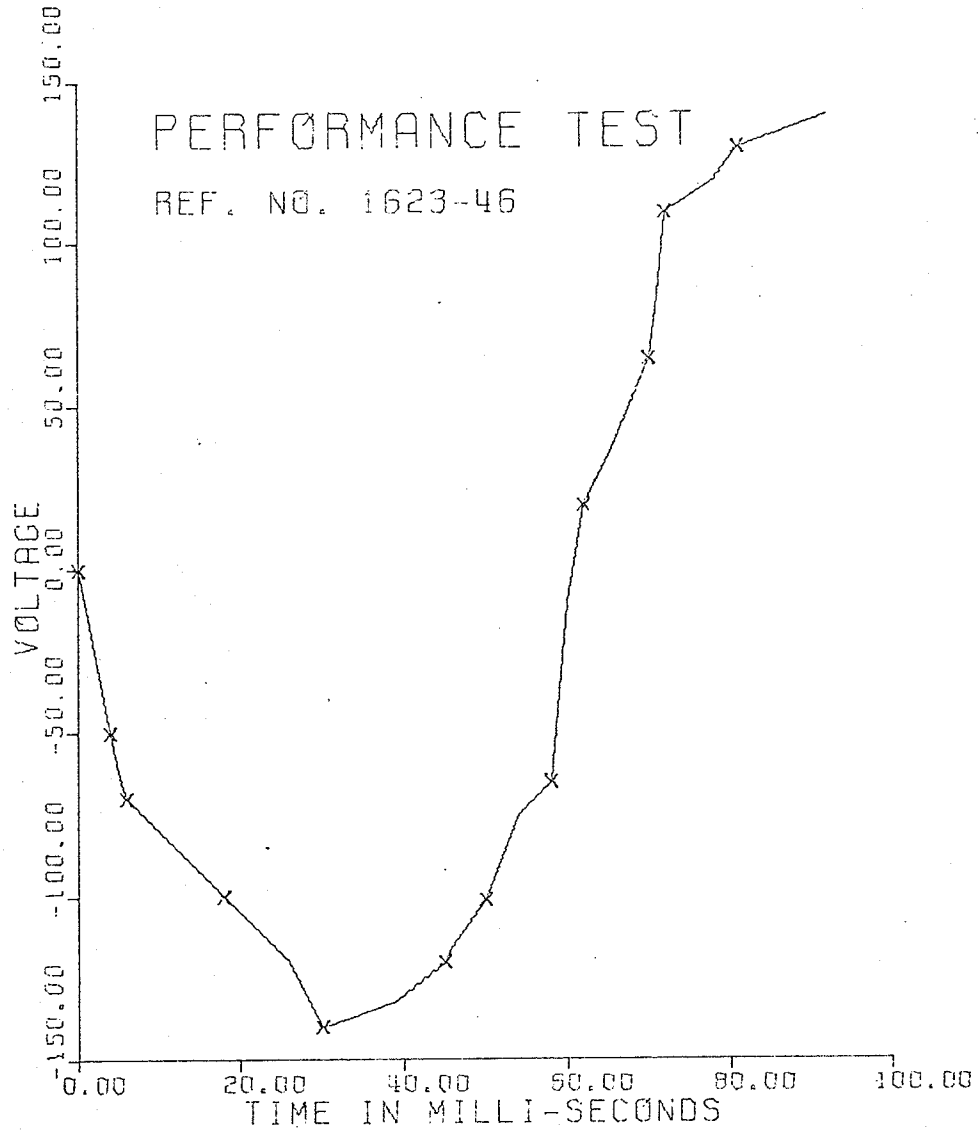
APPENDIX B

This appendix contains program listings of PLOTS, SYMBOL, SIMBLE, NUMBER, SCALE, AXIS, AXISI, and LINE.

APPENDIX C

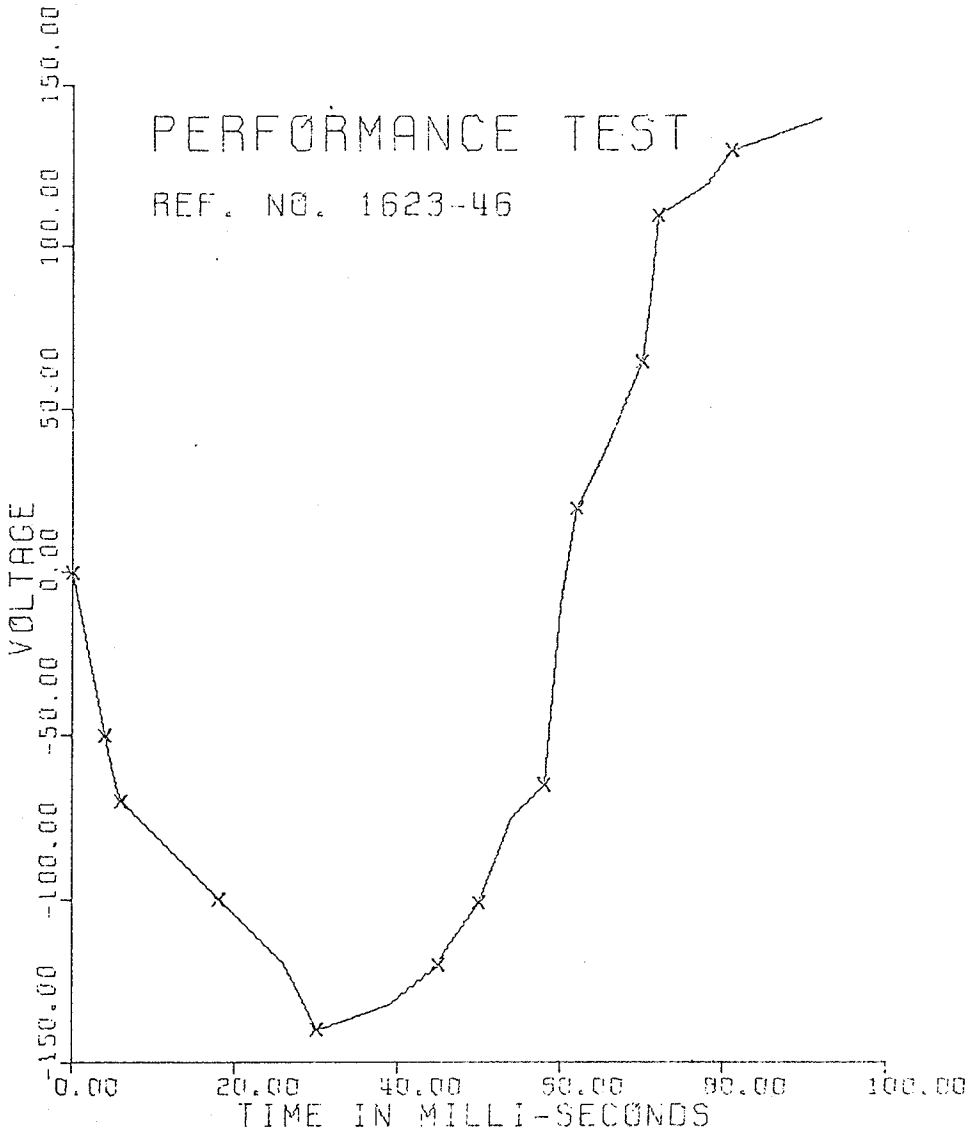
This appendix contains a listing of a sample program run using both the WATFIV and FORTRAN G compilers. The identical plot produced by these programs is also included.

```
C
C  THE FORTRAN G VERSION OF THE CALCOMP PLOT ROUTINES
C
0001        DIMENSION IBUF(4000),XARRAY(26),YARRAY(26)
0002        CALL PLOTS(IBUF,4000)
0003        READ 25,(XARRAY(I),YARRAY(I),I=1,24)
0004     25 FORMAT(2F6.2)
0005        CALL PLOT(0.,-5.,-3)
0006        CALL PLOT(0.,.5,-3)
0007        CALL SCALE (XARRAY,5.,24,1)
0008        CALL SCALE (YARRAY,6.,24,1)
0009        CALL AXIS(0.,0.,21HTIME IN MILLI-SECONDS,-21,5.,0.,XARRAY(25),
           1XARRAY(26))
0010        CALL AXIS(0.,0.,7HVOLTAGE,7,6.,90.,YARRAY(25),YARRAY(26))
0011        CALL LINE(XARRAY,YARRAY,24,1,2,4)
0012        CALL SYMBOL(.5,5.6,0.21,16HPERFORMANCE TEST,0.,16)
0013        CALL SYMBOL(0.5,5.2,0.14,16HREF. NO. 1623-46,0.,16)
0014        CALL PLOT(12.0,0.0,999)
0015        STOP
0016        END
```



MADE IN CANADA

```
      $JOB  WATFIV
      C
      C   THE WATFIV VERSION OF THE CALCOMP PLOT ROUTINES.
      C
 1          DIMENSION IBUF(4000),XARRAY(26),YARRAY(26)
 2          CALL PLOTS(IBUF,4000)
 3          READ 25,(XARRAY(I),YARRAY(I),I=1,24)
 4       25 FORMAT(2F6.2)
 5          CALL PLOT(0.,-5.,-3)
 6          CALL PLOT(0.,.5,-3)
 7          CALL SCALE (XARRAY,5.,24,1)
 8          CALL SCALE (YARRAY,6.,24,1)
 9          CALL AXIS(0.,0.,21HTIME IN MILLI-SECONDS,-21,5.,0.,XARRAY(25),
           1XARRAY(26))
10          CALL AXIS(0.,0.,7HVOLTAGE,7,6.,90.,YARRAY(25),YARRAY(26))
11          CALL LINE(XARRAY,YARRAY,24,1,2,4)
12          CALL SYMBOL(.5,5.6,0.21,16HPERFORMANCE TEST,0.,16)
13          CALL SYMBOL(0.5,5.2,0.14,16HREF. NO. 1623-46,0.,16)
14          CALL PLOT(12.0,0.0,999)
15          STOP
16          END
```



MADE IN CANAD

APPENDIX D

REFERENCES

1. Cress, P., Dirksen, P., and Graham, J. W. FORTRAN IV with WATFOR and WATFIV. Prentice-Hall Inc. Englewood Cliffs, N. J. . Revised 1970.

2. Cress, P., Dirksen, P., and Ward, S. J. /360 WATFIV Implementation Guide. Dept of Applied Analysis and Computer Science, University of Waterloo, Waterloo, Ontario. Revised September, 1969. 84-92.

3. IBM Corporation: IBM Systems Reference Library: OS Assembler Language Release 21 Form No. GC28-6514-8, IBM Nordic Laboratory, Publications Development, Sweden. Revised January, 1972.

4. IBM Corporation: IBM Systems Reference Library: IBM System/360 Operating System: FORTRAN IV (G and H) Programmers Guide Form No. GC28-6817-3. Programming Publications, N.Y. . Revised September, 1972.

5. IBM Corporation: IBM Systems Reference Library: IBM System /360 Operating System JOB Control Language Reference Form No. GC28-6704-3, Publications Development, N.Y. . Revised April, 1973.

6. IBM Corporation: IBM Systems Reference Library: IBM System /360 Principles of Operation Order No. GA22-6821-8 Systems Development Division, N.Y. . Revised November, 1972.

7. University of Manitoba Computer Reference Manual: PLOTTING ON THE UNIVERSITY OF MANITOBA SYSTEM/360 CALCOMP 750/563 INCREMENTAL DRUM PLOTTER. Revised March, 1972.

8. Calcomp Basic Software Plotting manual: Programming Calcomp Pen Plotters. California Computer Products Inc. , California. June 1968.

9. University of Waterloo reference Manual: A Guide to Using the University of Waterloo Level G Assembler for the IBM /360. Waterloo,Ontario. Seventh edition, revised August, 1971.

10. University of Waterloo Reference Manual: Calcomp Functional Software. Computer Centre, Waterloo, Ontario. Copyright 1970.