Computing Centre

University of Waterloo
Waterloo, Ontario
Canada N2L 3G1

A GUIDE TO USING THE UNIVERSITY OF WATERLOO LEVEL G ASSEMBLER FOR THE IBM SYSTEM/360 OR SYSTEM/370

PROJECT MEMBERS

MIKE DOYLE
RENNIE PETERSEN
STEVE SCHROETER

DAVE POTTER
RODNEY COOPER
BRUCE UTTLEY

TENTH EDITION

JUNE 1976

University of Waterloo
Waterloo, Ontario.

PRINTED IN CANADA

DISCLAIMER

ACKNOWLEDGEMENTS

ASMG is a modification to IBM'S level (F) Assembler IEUASM. Extensive use was made of the program logic manual for IEUASM and we are indebted to the writers and documentors who provided such a clear description of such a very large program.

The changes to ASMG to permit it to build larger local dictionaries (and thus assemble larger programs than Assembler (F)) are due to Christine Packard and George Sjoberg of the Pennsylvania State University Computation Center.

The changes to ASMG to support named common, and the optional support for the Model 67 RPQ instructions are due to Martin Raim of the University of Michigan Computing Center.

The part of ASMG which determines the day of the week for printing on the heading page was inspired by, and is somewhat modelled after, the program 'WEEKDAY' written by Richard L. Conner.

The alternate root phase for the Assembler called ASMGWYL which supports WYLBUR format input files was adapted from code written by Andrew Koenig of Columbia University.

TABLE OF CONTENTS

Introduction. 5

Assembler Options 6

Data Sets Required 13

Batch Option 15

Execute Option 16

Batch and Execute Option 17

Update Option 17

Exten Option 18

Assembler Output 22

Invoking ASMG 28

Appendix A - ASMG Error Messages. 29

Appendix B - ASMG Instruction Sets. 52

INTRODUCTION

ASMG is a modification to the OS/360 level (F) Assembler. The project was undertaken at the University of Waterloo in the summer of 1967 in the hope of attaining five basic aims.

A) To extensively reduce the time taken for an assembly.

B) To provide a batch-processor for student and other small assemblies.

C) To produce a change in the format of the cross reference dictionary by generating the same information as before but in many fewer pages.

D) To allow suppression of the external reference dictionary and the relocation dictionary.

E) To allow the selection of optional instruction sets to be recognized by the assembler.

These basic aims have been successfully completed.

In addition, ASMG has since been modified to permit it to assemble larger programs than Assembler (F) can by allowing the unsubsetted dictionary area to be greater than 64K.

Concatenation of unlike datasets or datasets on unlike devices is supported for SYSIN and the optional SYSUP.

No restrictions have been made to the assembler language and the input to ASMG may be the same as for Assembler (F). Several language extensions have been added to Assembler (G), but all under control of an EXTEN option. Thus strict conformity with the language rules of Assembler (F) may be maintained if desired. See the section on the EXTEN option.

Since most of the differences between ASMG and Assembler (F) are internal we will attempt herein only to describe those functions normally discussed in a programmer's guide. Unless specifically stated, ASMG does not differ from the OS/360 level (F) Assembler as described in the IBM Assembler (F) Programmer's Guide. * A working knowledge of this manual is assumed. This documentation corresponds to V2L7a of ASMG, which is roughly equivalent to Release 21.8 of Assembler (F).

---

* IBM System /360 Operating System Assembler (F) Programmer's Guide form C26-3756.

ASSEMBLER OPTIONS

The programmer may specify the following assembler options in the PARM= field of the EXEC job control card. The syntax rules governing the field are given in the OS/360 Job Control Language manual.

The assembler options specified in the OS level (F) Assembler programmer's guide are still permitted and have the same (or similar) functions as before. The only differences are that LOAD,NODECK is the default instead of DECK,NOLOAD, and that the parameter XREF causes a compressed format XREF to be printed. The Assembler (F) XREF may be requested via the parameter FULLXREF.

The entries may appear in any order and if any are missing a standard default will be assumed. Blanks and/or commas are accepted as parameter delimiters and keyword operands may be delimited on the left with equal signs or may be surrounded by parentheses.

Upper case letters indicate the minimum length required to uniquely identify the parameter. Any alternate form of the parameter is enclosed in parentheses following the preferred form. When an option is underlined, this indicates that it is the default choice for this option.

When a numeric quantity is specified (for example in EXTIME= parameter) the following rules hold -- leading zeros are permitted, maximum number of digits is 7, and the number may be followed by the letter K, in which case the multiplicitive factor 1024 is applied to the number.

ALgn (ALIgn)	The assembler flags all alignment errors that it detects.
NOALgn (NOALIgn)	The assembler only flags alignment errors that involve the fetching of an instruction (E.G. branches or LPSW).
NOBatch (NOMULt)	After processing one deck, the assembler returns to the system.
Batch (MULt)	The assembler assumes there are many decks in SYSIN to be processed, each delimited by an 'END' card. See the section on the BATCH option.
CAlign=	This parameter may be used to set the column in which the comment field of generated statements is to appear. An arithmetic value from 0 to 255 is valid with the default value of 0 meaning the comment field is to appear starting in the column the original comment started. If the desired column is occupied by some other field, the comment will start one blank after the end of the operand field. The first line of a continued statement is considered as columns 1 to 71, the second line as 72 to 127 starting in column 16 and so on.

CMS	If ASMG is running on a CMS machine this option is default. If specified on a non-CMS machine the effect is to disable the incore macro table construction and lookup which will increase macro edit time and available core at the expense of performance.
COlumn=	Specifies the number of vertical columns in which the RLD, UMAP and FULLXREF or XREF listings are to appear.

COL=0	UMAP in 1, XREF in 1, RLD in 3.
COL=1	UMAP in 1, XREF in 1, RLD in 1. This is the default.
COL=2	UMAP in 2, XREF in 2, RLD in 2.
COL=3	UMAP in 2, XREF in 2, RLD in 3.

Deck (PUNch)	The object module is placed on the device specified in the SYSPUNCH DD statement.
NODeck (NOPUNch)	No object deck is punched. A SYSPUNCH DD card is not required.
NOESd	The external symbol dictionary is not printed.
ESd	The external symbol dictionary is printed on SYSPRINT.
NOEXECute	The assembler does not attempt to load and execute the object module.
EXECute	The assembler attempts to load and execute the object module. See section on the EXECUTE option.
EXten	Certain extensions to the assembler language supported by Assembler (F) are allowed. See the section on the EXTEN option.
NOEXten	Strict language compatability with the OS (F) Assembler is observed. To disable the extended branch conditional register instructions and PUSH/POP you should also specify INSTSET=0.
EXTime=	Execution time allowed each job executed under the EXECUTE option. The value may range from 1 second to 9999 seconds. The default is EXTIME=5. The EXTIME= option is only meaningful if EXECUTE is also specified.

INSTset= (ISet=)	The instruction set which the assembler recognizes is set by this parameter. See appendix B for the total OP-CODE table. The default is INSTSET=1, an alias for IS=70.
IS=0	Instruction set compatible with OS Assembler (F).
IS=9	Instruction set compatible with DOS Assembler (F).
IS=20	Same as IS=60 except all non model 20 instructions removed and the model 20 only instructions added.
IS=44	Same as IS=20 except oriented toward the model 44.
IS=60	Same as OS Assembler (F) except that the extended branch conditional register mnemonics are added and all 370 instructions are removed.
IS=67	Same as IS=20 except oriented toward the model 67.
IS=70	Same as OS Assembler (F) except that the extended branch condition register mnemonics are added.
IS=71	Same as IS=70 with VM/CP instructions added.
LINECNt=nn (LINEcount=)	This parameter specifies the number of lines to be printed between headings in the listing. The limits are from 0 to 254 lines per page. Note that 0 lines means an infinite number of lines per page except when EJECT or its equivalent is encountered. The default is LINECNT=55.
List	A source listing of the programmer's macros, source cards and copied cards are printed on SYSPRINT (under control of PRINT statements).
NOList	No source listing is printed. The SYSPRINT DD card is still required unless the TERM option is specified and a SYSTERM DD card is provided.
FULLList (FList)	As well as the usual listing produced by the LIST option, a listing is given of all the library macros used by the program. This listing follows the normal listing. An internally generated TITLE statement preceeds each macro.
LOad (OBJect)	An object module is written on the data set specified by SYSLIN. If SYSLIN is not present but SYSGO is, then SYSGO will be used instead.
NOLOad (NOOBJect)	No object module is written on the data set specified by SYSLIN. A SYSLIN card is not required.
LRef	A literal cross-reference is printed. The format of this output is described in the section titled 'Assembler Output'.
NOLRef	No literal cross-reference is printed.

LSetc=	This parameter may be used to set the default length of a SETC variable, GBLCs and LCLCs, to a value other than 8. The value may range from 1 to 255 bytes. Space for SETC variables is allocated in a static fashion so a value less than 8 will save core in the conditional assembly but a value greater than 8 will require extra core. If NOEXTEN is on then 8 will always be used.
Number	The line number field (columns 73-80) is written on SYSTERM for statements for which diagnostic information is given. This option is considered only in connection with TERM.
NONumber	No line number field (columns 73-80) is written on SYSTERM for statements for which diagnostic information is given.
OS	The assembler does not attempt to be compatible with the DOS (F) Assembler.
DOS	Q and L type constants will be flagged as errors and the RLD will not be sorted. For complete compatability with the DOS (F) Assembler you should also specify NOEXTEN and INSTSET=9. DOS may be used together with any other options.
PRinter (PRT)	The Assembler (G) heading page and the trailing diagnostics will be listed on SYSPRINT. All other listing segments operate under their own parameter control.
NOPRinter (NOPRT)	The SYSPRINT data set is not opened. All listings and diagnostics are lost. The TERM option must also be specified and a SYSTERM DD card provided.
Rent	The assembler checks for a possible coding violation of programme reenterability.
NORent	No programme reenterability checking is done.
NORLd	The relocation dictionary is not printed.
RLd	The relocation dictionary is printed on SYSPRINT.
SPace=	The SPACE parameter specifies how much main storage the assembler should attempt to use for I/O buffers (excluding QSAM, (SYSIN, SYSPRINT, SYSLIN, SYSPUNCH, SYSTERM and SYSUP buffers), tables (macro directory, macro generation dictionaries, assembly symbol table, and XREF and RLD tables) and the execution time load area (if EXECUTE option specified)). The space parameter does not include the following -- The assembler itself (about 30K), the access methods (2K-5K depending on number resident in the system), the QSAM buffers (size dependent on user datasets), the instruction set module (about 2K, only if CMS, MFT, PCP or VS1, 4K for MVT or VS2). The SPACE parameter may be coded in three different ways --

SPace=NNN	NNN is a number representing the amount of storage the assembler is to use. Minimum is 12736 bytes. For example SPACE=200K.
SPace=MAX	This tells the assembler to use all available storage. Use of this form of the space parameter is not recommended.
SPace=MAX-NNN	NNN is the number of bytes of storage the assembler leaves free for OS. This storage is also available for use by the executed program (if EXECUTE is specified) or for extra SYSIN or SYSUP buffers if unlike datasets are concatenated on SYSIN or SYSUP. For example SPACE=MAX-20000.

The default is -- SPACE=MAX-2K For MFT, PCP or VS1.
SPACE=MAX-4K For CMS, MVT or VS2.

SImt	The statement number is written on SYSTERM for statements for which diagnostic information is given. This option is considered only in connection with TERM.
NOSTmt	No statement number is written on SYSTERM for statements for which diagnostic information is given.
SYsparm=	This is one of the EXTEN options. It specifies the character string value of the System Variable Symbol &SYSPARM. If the SYSPARM= option is not coded then &SYSPARM will default to a null character string. If EXTEN is not on then &SYSPARM may not be referenced, and any SYSPARM= parameter on the EXEC card will be ignored.

Commas are not allowed unless parentheses or quotes surround the entire SYSPARM value. Two quotes and two ampersands are needed to represent one.

```
e.g., PARM=(LOAD,'SYSPARM=(&&AB,(''&&XY))')
                    returns
           (&AB,('&XY)) to &SYSPARM.
```

TERminal	The assembler writes diagnostic information on the SYSTERM data set. Options NUM and STMT are meaningful only if TERM is specified.
NOTERminal	No diagnostic information is written out on the SYSTERM data set.
TEStran	The object module contains the special source symbol table required by the test translator (TESTRAN) routine and the TSO TEST processor.
NOTEStran	No testran symbol table is produced.
UMap	A Using Map of all registers in USING and DROP and POP USING statements is printed.
NOUMap	No USING Map is printed.

UPCond=	An arithmetic value from 1 to 20. The default is 12. Update diagnostics ASMG320 and following have been assigned an internal severity code. If the internal code exceeds UPCOND, the assembly or assemblies if BATCH will be terminated at the start of macro expansion with an ASMG115.
UPDate	An update deck (SYSUP) and an old master data set (SYSIN) are read simultaneously. The assembly will be done on the resulting (non-existant) new master. The SYSUP data set may contain ./ DELETE, ./ NUMBER, ./ ENDUP and ./ * (Comment) control cards in addition to sequenced update cards. Other control cards are ignored.
NOUPDate	The SYSIN data set only is assembled.
UPList	Changes from SYSUP are listed in the Update Log on SYSPRINT before the source listing.
NOUPList	No changes from SYSUP are listed on SYSPRINT.
FULLUPList (FUplist)	The non-existent new master from SYSUP and SYSIN is listed on SYSPRINT.
UTbuff= (UBuff=)	The number of utilities which the assembler attempts to buffer in core. For most programmes UB=3 is best. For extremely large programmes UB=1 should be used. The cutoff point between extremely large and other programmes depends on the memory size. UB=0 should only be used when memory is exceeded during assembly. The default is UTBUFF=3.
UTBUFF=0	No utility is buffered in core.
UTBUFF=1	SYSUT1 is buffered in core.
UTBUFF=2	SYSUT1 and SYSUT2 are buffered in core.
UTBUFF=3	SYSUT1, SYSUT2 and SYSUT3 are buffered in core.
FULLXref (FXref)	The assembler produces a cross-reference table of symbols as part of the listing.
Xref	A condensed cross-reference table of the symbols used in the programme is printed on SYSPRINT. This format is described in the section titled 'Assembler Output'.
NOXref	No symbol cross-reference table is produced.
Note --	Any of the three XREF parms may be used as a keyword with FULL or SHORT as the operand. XREF=SHORT removes all entries from the symbol cross-reference table that are defined but never referenced. XREF=FULL produces a cross reference table of all symbols used in the assembly. FULL is the default and is implied if no operand is present for one of the XREF parms.
YFlag	Diagnostic message ASMG046 and its severity may appear in the listing.
NOYFlag	Diagnostic message ASMG046 and its severity is always suppressed.

The following is an example of specifying assembler options --

```
//      EXEC     PGM=ASMG,PARM='LOAD,UB=2,NOES'
```

Should two parameters reference the same function, the last mentioned is used, except for SPACE= which is unpredictable. A null parameter will be bypassed without causing an error.

If no options are specified the PARM FIELD will be assumed as --

```
PARM='ALGN, NOBATCH, CALIGN=0, COLUMN=1, NODECK, NOESD, NOEXECUTE,
      EXTEN, EXTIME=5, INSTSET=1, LINECNT=55, LIST, LOAD, LREF,
      LSETC=8, NUM, OS, PRINT, NORENT, NORLD, SPACE=MAX-4K, STMT,
      SYSPARM=, NOTERM, NOTEST, UMAP, UPCOND=12, NOUPDATE, UPLIST,
      UTBUFF=3, XREF, YFLAG.
```

The space option will be MAX-2K if the system is MFT, PCP or VS1.

DATA SETS REQUIRED

DD cards must be present for SYSIN and the three utility datasets. All other DD cards are optional, being controlled by parameters with the exception of SYSLIB. The absence of any necessary DD card will be detected and an error message typed for SYSPRINT, unless SYSTERM is open, and printed for all others. Datasets required are compatible with Assembler (F) except that SYSLIN is the preferred ddname for the object output dataset (SYSGO is accepted if it is used instead of SYSLIN). Also, the default BLKSIZE for SYSLIN and BUFNO for SYSPUNCH and SYSLIN are different.

SYSIN — Blocksize must be a multiple of 80 and is defaulted at 80. LRECL is fixed at 80 and BUFNO default is 2. SYSIN may be concatenated and the datasets need not have like attributes nor reside on like devices. However, if any subsequent concatenation requires larger buffers than the first, then PARM='SPACE=MAX-10K' should be coded on the EXEC card to ensure that enough storage will be available for the buffers, or the first concatenation should specify DCB=BLKSIZE=largest.

If the first load of the Assembler is PGM=ASMGWYL, then the SYSIN file may be a WYLBUR Edit format file with a 'U' RECFM.

SYSLIB — Needed only if system macros or copy code used by the assembler source. Blocksize must be a multiple of 80 and is defaulted at 80. If datasets with different blocksizes are concatenated on SYSLIB, then the DD card with the largest blocksize should be first, or the first DD card should specify DCB=BLKSIZE=largest on it. Other than different blocksizes, datasets concatenated on SYSLIB must have like characteristics and reside on like devices.

If the first load of the Assembler is PGM=ASMGWYL, then the SYSLIB file may include concatenations of WYLBUR Edit format files with 'U' RECFM.

SYSPRINT — Blocksize must be a multiple of 121 or 133 and is defaulted at 121. BUFNO default is 2. The SYSPRINT DD card is not required if NOLIST,TERM options are specified and a SYSTERM DD card is present.

SYSPUNCH — Needed only if DECK is specified. Blocksize must be a multiple of 80 and is defaulted at 80. BUFNO default is 3.

SYSLIN/SYSGO — Not needed if NOLOAD is specified. Blocksize must be a multiple of 80 and is defaulted at 3200. BUFNO default is 2.

SYSUT1,2,3 Blocksize is accepted from a DD card and must be 1608 or larger. If the blocksize is not specified on the DD card then the assembler sets the blocksize as follows -- If SPACE<48K then UT1=UT2=UT3=1608, otherwise UT1=1608+(SPACE-48K)/16, UT2=UT3=(SPACE-48K)/8. These values are then rounded down to a double word multiple of a blocksize that will fit on one track with minimum wastage or 1608, whichever is greater. Blocksize values, if any, present in the dataset label are ignored.

In general, it is recommended that the user not specify a BLKSIZE on the utility DD cards, thus permitting the assembler to chose a suitable value. However, by specifying the minimum value it may be possible to permit a large assembly to run in less core, or by specifying a large value it may be possible to improve performance somewhat.

SYSUT1 must be on a direct access device. SYSUT2 and SYSUT3 may be on a direct access device, but if the utility is not buffered (UB<3 for SYSUT3 or UB<2 for SYSUT2) then the utility may be on tape. None of the three utilities may be concatenated.

SYSTERM Needed only if the TERM option is specified. Blksize must be a multiple of 121 or 133 and is defaulted at 121. BUFNO default is 2.

SYSUP Needed only if the UPDATE option is specified. Blksize must be a multiple of 80 and is defaulted at 80. BUFNO default is 2. SYSUP may be concatenated. See SYSIN for the rules.

If the first load of the Assembler is PGM=ASMGWYL, then the SYSUP file may be a WYLBUR Edit format file like SYSIN.

BATCH OPTION

If the option BATCH is specified (without EXECUTE), the assembler will accept multiple source decks in the SYSIN dataset. The decks are only delimited by the END card of the preceding deck, and the end of the batch is signaled by end-of-file on SYSIN. The listing of each source deck is preceded by the usual header page and the object decks appear one after another on SYSPUNCH and SYSLIN. Missing or erroneous DD cards are only flagged on the listing of the first source deck.

A Batch Summary will terminate the SYSPRINT listing. This summary will number the assemblies sequentially from one, list the name field from the first TITLE card of each assembly, how many errors were detected in each assembly and what the highest severity code was for each assembly. The number of errors in the batch and the highest severity code for the batch is listed last. The condition code returned to the invoking programme (usually OS) is the highest code encountered in any of the assemblies in the batch.

When the batch option is used the assembler requires about 8K more storage than without the batch option.

Note that MULT is a synonym for BATCH and NOMULT is a synonym for NOBATCH.

EXECUTE OPTION

When the EXECUTE option is specified the object module is written out on SYSUT2. It is then read in by phase FEX of ASMG and loaded into core. FEX prints a load map showing the actual location each CSECT is loaded into. FEX then executes a SPIE to get control of programming interrupts, and a STIMER to get control back if the programme exceeds its allotted execution time, possibly because of an infinite loop. Finally it points R13 at a new save area, and points R1 at LIST1 in ASM and does a BALR R14,R15 to the programme's entry point. If an interrupt from the SPIE or STIMER occurs, FEX gives a memory dump of the users core. Any previous SPIE will be retained and reinstated by FEX.

The time used in the STIMER macro is the time specified by the EXTIME= option in the PARM FIELD.

The user is allowed to do I/O provided he supplies his own DD cards and does his own OPEN'S, or he may use the assembler's DD cards subject to the following restrictions. Since the assembler's datasets are still open, the assembler's DCBs must be used in order to do I/O on the assembler's DD cards.

If the user wishes to use the assembler's datasets he can find the addresses of the DCB'S through register 1. However, he should not do I/O on any utility which is buffered (I.E. if UTBUFF=2, then SYSUT1 and SYSUT2 should not be used). The DCB addresses are 4 bytes each and are in the order given in the section 'Invocation of ASMG.' Register 1 points at the first one (example address of SYSPRINT DCB is at (R1)+20). The data set RECFMs and LRECLs are the same as described in the Assembler (F) Programmer's Guide with the following exceptions. SYSPRINT and SYSTERM have a default LRECL of 121 but 133 may be specified as an alternative. All non-utility output data sets should be written with PUT MOVE conventions. All non-utility input datasets should be read with GET MOVE conventions. SYSPRINT should be written with ASA carriage control characters.

If SYSIN is used and EOF is read, a message is printed and execution is terminated. The user may alter the SYSIN DCB'S EODAD entry if he wishes.

The amount of free storage available to the user's program depends on the SPACE option.

If SPACE=MAX was specified there is no free storage.

If SPACE=nnn was specified then the available storage is the problem program partition or region size minus nnn minus the assembler code minus some miscellaneous other things.

If SPACE=MAX-nnn was specified then nnn bytes of free storage should be available.

See explanation of the SPACE parameter.

Since FEX is a simple one-pass loader the following things are not supported -- EXTRN statements, V-type address constants, and pseudo-register vectors. Multiple CSECT'S and COM is supported.

BATCH AND EXECUTE OPTION

If both BATCH and EXECUTE are specified, then each source deck must be preceded by a card which contains $JOB in columns one to four, and a blank in column five. This card is printed on the heading page. Each programme is executed immediately after its assembly. The $JOB card appears as an end-of-file to the assembler and to the user's programme if he uses SYSIN. Real end-of-file still signals the end of the batch.

In this case no Batch Summary will terminate the SYSPRINT listing.

UPDATE OPTION

If the Assembler (G) Update facility is specified by PARM=UPDATE, then a SYSUP DD card is necessary. The update deck must contain sequence numbered update cards. Any ./ control cards other than DELETE, NUMBER, ENDUP or '*' are ignored. The SYSIN deck must also be correctly sequenced.

For compatibility with IEBUPDAT the 'DELET' function and its operands are equivalent to 'DELETE' and 'CHNGE' is equivalent to 'CHANGE'. As extensions a './ *' function is recognized as a Comment and SEQ2 may be omitted taking SEQ1 as its default value.

A ./ ENDUP card signals end-of-file on the SYSUP data set. If more records do exist then an UNPROCESSED SYSUP RECORDS error (ASMG340) will be produced. A real end-of-file on SYSUP also terminates SYSUP processing.

An update log will preceed the assembly depending on the UPLIST parm specified. If EXECUTE is specified and the user reads on SYSIN, then the update log may get printed among the user's output. BATCH will work properly if the sequence numbers between decks are strictly increasing.

The UPCOND= keyword parameter sets a condition code that the UPDATE option tests. If the UPCOND= value is exceeded the assembly aborts at the start of macro expansion with an ASMG115 diagnostic. UPDATE severity codes have been assigned to ASMG320 and following with the following meanings --

```
 0 -- just a comment
 4 -- an interesting comment
 8 -- unsupported function
12 -- possible error, maybe all ok
16 -- probable user error
```

Severity 8 and higher are listed on SYSTERM if TERM.

Using the UPDATE facility will increase the region required for ASMG by 4K plus the size of the SYSUP buffers.

EXTEN OPTION

The EXTEN option permits several extensions to the language as supported by the OS Assembler (F). If compatability with Assembler (F) is desired, these extensions may be disabled by specifying PARM=NOEXTEN on the EXEC card. The following is a description of each of the extensions.

1) PRINT statements are permitted in macros. This can be used to write macros such that the outer macros will assemble under PRINT GEN, while the inner macros are under PRINT NOGEN. By having one or more GBLC'S as the operands of the PRINT statements, central control could be maintained over the PRINT statements in the macros.

2) The attributes of labels defined in an outer macro and passed to an inner macro as a parameter are available, subject to the usual restrictions on the statements that the labels are on. This differs from Assembler (F), which only keeps the attributes of labels defined outside of macros.

3) The system variable symbols &SYSNDX and &SYSECT may be used in open code as well as in macro definitions. Four additional system variable symbols are available for reference within macros and open code. These are --

&SYSDATE	This symbol contains the date of the assembly. It consists of six or seven characters in the following format - YYMONDD or YYMOND. For example, &X SETC '$&SYSDATE' could return '$70JAN15' or '$70FEB3'.
&SYSTIME	This symbol contains the time of the assembly. It is in the format HH:MM:SS. &SYSTIME remains constant for the whole assembly, but will change for each deck in a batch assembly. As an example &Y SETC '&SYSTIME' could return '16:34:19'.
&SYSSTYP	This symbol contains the type of section the macro was invoked in. Its contents will be one of 'CSECT', 'DSECT' or 'COM'. It remains constant across the expansion of one macro. It can be used to restore the assembly to the

section it was in when the macro was invoked.
For example --

```
                 MACRO
                  ..
                  ..
                 AIF    ('&SYSSTYP' EQ 'DSECT').DSECT
         &SYSECT  CSECT
                 MEXIT
         .DSECT   ANOP
         &SYSECT  DSECT
                 MEND
```

&SYSPARM This symbol contains the character string specified by the SYSPARM= parameter in the PARM field. It has a default value of a null character string (i.e. its length is zero).

4) Named common is supported (i.e. the COM statement may be labelled). This is useful when coding assembler subroutines for use with FORTRAN programs.

5) The maximum length of a SETC variable may be declared by the user on the LCLC or GBLC statement. The length declaration is specified by following the variable name by a '*' and the amount of storage to be allocated to the variable. For example --

```
          GBLC  &A*100,&B*1(256),&C
```

This declares &A to be a global SETC variable with a maximum length of 100 bytes, &B is an array of 256 global SETC variables, each having a maximum length of 1 byte (to save storage), and &C is a global SETC variable with a maximum length of 8 bytes (default). The length may be declared anywhere in the range from 1 byte to 255 bytes. All processing of SETC expressions is done using up to 255 bytes of data. When the assignment to the left side of a SETC statement is done, the expression is truncated on the right if necessary to fit into the SETC variable. All SETC variables have a current length, which is less than or equal to the declared maximum length, and which is initially zero null string). As part of this extension, the second expression of the substring notation may be as large as 255.

The default length of 8 bytes may be changed with the LSETC= parameter. The explicit declaration of a maximum length using '*' always overrides the default.

6) The K' (count) operator has been extended to allow any SETC variable as an argument. This extension may be used in conjunction with SETC variables with a maximum length other than 8. From the example in 5) --

```
     &N        SETA  K'&A
```

might return a value from 0 to 100 to &N.

The K' (count) operator also allows any SETA or SETB variable as an argument. With a SETA argument the number of digits in the number is returned and with a SETB argument the value one is always returned.

7) SETC variables which contain C, X, or B type self-defining terms may be used in SETA expressions. For example, if &CHAR contains a single character, then the following statements could be used to determine its EBCDIC value --

```
&CSDT     SETC  'C''&CHAR'''
&VALUE    SETA  &CSDT
```

8) Code copied by COPY may contain MACRO, MEND or COPY statements. Thus, library macros may be copied in by COPY at the start of an assembly and be treated as programmer macros. COPY code within COPY code is valid up to five nesting levels.

9) Extended DROP is supported. A DROP instruction with no operand or the null operand drops all registers currently in use.

10) Extended EQU's are supported. The second operand is the length of the symbol from 0 to 65535. If it is to be referenced at macro expansion time it must be a self-defining term, i.e. decimal, B, C or X. The third operand is the type of the symbol that may be referenced by the T' operator. This third type operand must be a self-defining term in the range 0 to 255.

11) Labelled CNOP, labelled ORG and unlabelled DSECT are valid.

12) Comments may be generated. A generated operand may contain a blank character. The character string following the blank is treated as a comment.

13) Assembly phase, but not yet Conditional Assembly phase, arithmetic expressions may contain unary operators + and -, up to 11 levels of parenthesis instead of 5 and up to 25 terms instead of 16.

14) Eight character TITLE labels are valid. These are truncated to four for columns 73-76 of the object deck but elsewhere are maintained as eight.

15) Only one TITLE statement in an assembly may have a label but the label does not have to occur on the first TITLE statement.

16) Expressions are allowed in the Duplication and Length factors of a literal.

17) The Current Location Counter symbol '*' is allowed in Duplication and Length factor calculations for DCs, DSs and literals.

18) Support for four byte self-defining terms in the Assembly Phases.

19) Allow the dimension of a SETx variable to be up to 9999.

20) 'END' statement is allowed in COPY code not within a Macro. Any statements following the 'END' in the COPY member are treated as source comments. In 'BATCH' mode the next assembly will start with the first SYSIN statement following the 'COPY'.

21) An MNOTE with only a quited string for an operand is treated as a comment when printing.

22) The Positional and Keyword parameters may be intermixed in the Macro Prototype and Macro Instruction statements.

23) LCLx, GBLx and ACTR statements can appear anywhere before the variable's use in Open code or within a Macro definition.

ASSEMBLER OUTPUT

Unless otherwise specified the assembler output is identical with that discussed in the Assembler (F) Programmer's Guide. Within an assembly, page numbering is consecutive from one.

ASSEMBLER HEADER PAGE

Before the printing of the external reference dictionary (which is now optional and not printed unless requested by an ESD parameter in the EXEC job control card), several lines of information are printed out in double spaced format. The first line contains seven keywords.

LEVEL=G	This specifies the level of the assembler.
RELEASE=	The current release date in the form date month year with no separating blanks.
SYSTEM=	The type of OS that ASMG is running under - CMS, MFT, MVT, PCP, VS1 or VS2 plus the Release number of this version.
MODEL=	The two or three digit model number of the machine.
TIME=	The time of day when this assembly was done. This is in the form HOUR:MINUTE:SECOND and is in the standard 24 hour universal clock format. I.E. 00:11:00 would be 11 minutes after midnight whereas 13:11:00 would be 1:11 PM.
DAY=	The name of the day.
DATE=	The date, month and year separated by blanks.

The second line lists the OVERRIDING PARM specified on the EXEC card only if any was present. If the parm field contains an error an asterisk is placed under the start of the error. The rest of the the parm field after the error is ignored.

Then ASMG prints a list of the assembler options used to run this current job. These options are those specified on the EXEC job control card and/or the default options assumed.

UPDATE LOG

This section of the assembly listing logs errors and information about the UPDATE option. If the NOUPLIST option is specified then only errors and the associated card is listed. These update errors are also listed on SYSTERM if TERM is specified. The UPLIST option causes update action messages to be listed including deletions, insertions and replacements. FULLUPLIST lists the UPLIST records plus all those records passed to the assembler from SYSIN without change.

SOURCE LISTING

The date is printed in the form --
DATE MONTH YEAR separated by blanks.

Argument values of EQU, ORG and USING appear in the ADDR2 field.

Under PRINT NOGEN the first location counter value will print beside the last line of the open code macro call, if any location counter is generated by the macro call.

Both the previous location counter and the new location counter are printed for ORG statements.

Non-comment MNOTEs are flagged by having '***MNOTE***' appear to the left of the statement.

The comment field of generated statements starts in the same column as the original macro statement comment if possible, or one column to the right of a long generated operand if not. See also the CALIGN= parameter for other aligning possibilities.

When a line is in error, ASMG attempts to print the line even if it would not normally be printed due to PRINT OFF, NOGEN, and/or NODATA.

RELOCATION DICTIONARY AND EXTERNAL SYMBOL DICTIONARY

The RLD and ESD listings have not been changed except that they are currently not printed at all unless the parameters RLD and/or ESD are specified in the PARM= field of the EXEC job control card. See the COL= parameter to print the RLD in multiple columns.

THE USING MAP

This section provides the programmer with information about base registers used and dropped by USING, DROP and POP USING assembly statements. These references are listed in ascending order by register number.

REGISTER	A value from 0 to 15 indicating the register.
USING STMT	The statement number in which the USING or POP USING statement was issued.
DROP STMT	The statement number in which the DROP or POP USING was issued. 'END' appears in this field if a DROP was never issued.
VALUE	The address at which the base register was set.
LABEL	Is the first operand of a USING statement. If that operand is longer than twelve characters then the twelfth character is replaced by a period. If the register is being used as a result of POP USING then '*** POP ***' appears in this field.

THE LITERAL CROSS-REFERENCE DICTIONARY

This section provides the programmer with all the information about literals that the Cross-Reference dictionary does about symbols, under control of the LREF/NOLREF parameter. The format has been altered for ease of reading while allowing for the extreme and variable lengths of literal strings.

The format is of the form
LOCATION LENGTH DEFINITION LITERAL REFERENCES
where leading zeros are suppressed (except in LOCATION) and

LOCATION	Is the address at which the literal has been generated.
LENGTH	Is the length attribute of the literal. This will be the length in bytes of the field occupied by the literal location unless a replication factor has been used.
DEFINITION	Is the statement number of the statement where the literal is generated.
LITERAL	Is the source literal string defined as an operand. If the source literal string exceeds 100 characters in length then it will be truncated on the right to 100 characters.
REFERENCES	Are the statement numbers of the statements in which the literal appears as an operand. These are printed up to 14 references per line.

All literals will appear in the Literal Cross-Reference in the EBCDIC collating sequence of the character string that defines them.

A PRINT OFF listing control instuction or a NOLIST or NOXREF option on the EXEC card does not affect the production of the Literal Cross-Reference section of the listing.

THE CROSS-REFERENCE DICTIONARY

Major changes have been made to the symbol cross-reference dictionary. The same information is printed as before, but in a new format. Also see the COL= parameter for more format possibilities.

This new format limits excessive use of paper in cases where the cross-reference dictionary is only used by the programmer occasionally.

The old format can be obtained by specifying FULLXREF in the PARM= field of the EXEC job control card.

The new format is output as a 'stream' of data, and is of the form SYMBOL LENGTH,VALUE,DEFINITION REFERENCES SYMBOL LENGTH,... Where leading zeros are suppressed and

LENGTH	Is the length in bytes of the field occupied by the symbol value.
VALUE	Is the address the symbol represents or a value to which the symbol is equated.
DEFINITION	Is the statement number of the statement where the symbol is defined.
REFERENCES	Are the statement numbers of statements in which the symbol appears as an operand. In the case of a duplicate symbol, the assembler fills this column with the message -- *****DUPLICATE***** In the case of an undefined symbol -- *****UNDEFINED***** fills the length,value,definition field.

Symbols appearing in V-type address constants do not appear in the cross-reference listing.

A PRINT OFF listing control instruction or NOLIST option on EXEC card does not affect the production of the cross-reference section of the listing.

ASSEMBLER DIAGNOTICS

Assembler diagnotics are the same as those in the Assembler (F) Programmer's Guide with two major exceptions --

(1) All error messages are prefixed with the mnemonic ASMG instead of IEU and

(2) Several error messages have been added to aid in debugging BATCH/EXECUTE and UPDATE programmes, and to reflect the new data set and core management.

All ASMG diagnostic messages are listed in appendix A.

BATCH SUMMARY

If the BATCH and NOEXECUTE options are in effect, a BATCH SUMMARY page will be printed. See the description of the BATCH option for more information.

OBJECT DECK

The object deck produced by ASMG is the same as that produced by Assembler (F), except that ASMG may produce fewer TXT cards in the object module due to an improved packing algorithm. Also, the identification information placed in columns 33 to 51 of the END card is in the same format but contains unique identities. If no second IDR is present its space is occupied by the assembly date and time with the IDR count marked only one.

SYSTERM LISTING

A few changes have been made in the SYSTERM error listing.

The deck I.D. from a TITLE statement, the date and time are included in the ASSEMBLER (G) DONE message. Line numbers, printed under control of the NUM option, do not have high order zeroes suppressed. No list of assembly options is listed on SYSTERM.

If SYSTERM is routed to a printer with SYSPRINT, the SYSTERM listing will precede the SYSPRINT listing under HASP.

INVOKING ASMG

```
   <SYMBOL>  LINK     EP=ASMGASM,                                          X
                   PARAM=(OPTIONLIST<,DDNAMELIST<,ACMETHLIST>>),VL=1
or
   <SYMBOL>  ATTACH  EP=ASMGASM,                                           X
                   PARAM=(OPTIONLIST<,DDNAMELIST<,ACMETHLIST>>),VL=1
or
   <SYMBOL>  CALL     ASMGASM,                                             X
                   PARAM=(OPTIONLIST<,DDNAMELIST<,ACMETHLIST>>),VL
```

OPTIONLIST - Same as set up by OS for the PARM= FIELD of the EXEC card except that there is no limit on the length of the PARM.

DDNAMELIST - The first halfword contains the number of bytes in the remainder of the list. This remainder consists of 8 byte fields, each of which is all binary zeros or is a name left-justified and padded with blanks. Binary zeros indicate the use of a standard name. Entries may be omitted for names beyond the last one to be altered. The order of entries follows the ACMETHLIST description. Note SYSLIN and SYSTERM are in the list twice for Assembler (F) compatibility. Assembler (G) processes the overriding DDNAMES list in reverse order so that if SYSGO were in position 11 and SYSLIN were in position 1, SYSLIN would take effect but if position 1 were zeros SYSGO would take effect.

ACMETHLIST - A list of four byte entries specifying a code byte for an assembler dataset and a three byte address to be used as an access method routine. The end of the list is indicated with the X'80' bit on in the last code byte. Those data sets that are so overridden are not opened by ASMG, so actual DD declarations are not required or if present may be used by the invoker. Output exits should ensure the dummy DCB pointed to by (R1) has a valid LRECL to be used by subsequent PUT LOCATE move of data.

	DDNAME	Code Byte	I/O Conventions for ACMETHLIST
1	SYSLIN	X'00'	QSAM PUT LOCATE
2	SYSTERM	X'04'	QSAM PUT LOCATE
3	SYSUP	X'08'	QSAM GET LOCATE
4	SYSLIB	X'0C'	** not supported **
5	SYSIN	X'10'	QSAM GET LOCATE
6	SYSPRINT	X'14'	QSAM PUT LOCATE
7	SYSPUNCH	X'18'	QSAM PUT LOCATE
8	SYSUT1	X'1C'	** not supported **
9	SYSUT2	X'20'	** not supported **
10	SYSUT3	X'24'	** not supported **
11	SYSLIN	X'28'	QSAM PUT LOCATE
12	SYSTERM	X'2C'	QSAM PUT LOCATE

APPENDIX A

ASMG DIAGNOSTIC MESSAGES

ASMG DIAGNOSTIC MESSAGES

MESSAGE NUMBER		SEVERITY CODE
ASMG001	DUPLICATION FACTOR ERROR.	12
	A duplication factor is not an absolute expression, or is zero in a literal; * in duplication factor expression (valid if EXTEN); invalid syntax in expression.	
ASMG002	RELOCATABLE DUPLICATION FACTOR.	12
	A relocatable expression has been used to specify the duplication factor.	
ASMG003	LENGTH ERROR.	12
	The length specification is out of permissible range or specified invalidly; * in length expression (valid if EXTEN); invalid syntax in expression; no left-parenthesis delimiter for expression.	
ASMG004	RELOCATABLE LENGTH.	12
	A relocatable expression has been used to specify length.	
ASMG005	S-TYPE CONSTANT IN LITERAL.	8
	S-Type address constants may not be specified in a literal.	
ASMG006	INVALID ORIGIN.	12
	The location counter has been reset to a value less than the starting address of the control section; ORG operand is not a simply relocatable expression or specifies an address outside the control section.	
ASMG007	LOCATION COUNTER ERROR.	12
	The location counter has exceeded 2**24-1, or passed out of control section in negative direction (3 byte arithmetic).	
ASMG008	INVALID DISPLACEMENT.	8
	The displacement in an explicit address is not an absolute value within the range of 0 to 4095.	
ASMG009	MISSING OPERAND.	12
	Statement requires an operand entry and none is present.	

ASMG DIAGNOSTIC MESSAGES

MESSAGE NUMBER — SEVERITY CODE

ASMG010 INCORRECT SPECIFICATION OF REGISTER OR MASK. 8

One of the following --

1. The register or mask field specification is not an absolute value within the range 0-15.
2. An odd register is specified where an even register is required (multiply, divide, shift double instructions and move/compare long.)
3. The register specified was not a floating point register (for floating point instructions) or it was not an extended precision floating point register (for extended precision floating point instructions).

ASMG011 SCALE MODIFIER ERROR. 8

The scale modifier is not an absolute expression or is too large; negative scale modifier for floating point; * in scale modifier expression; invalid syntax or illegally specified scale modifier.

ASMG012 RELOCATABLE SCALE MODIFIER. 8

A relocatable expression has been used to specify the scale modifier.

ASMG013 EXPONENT MODIFIER ERROR. 8

The exponent is not specified as an absolute expression or is out of range; * in exponent modifier expression; invalid syntax; illegally specified scale modifier.

ASMG014 RELOCATABLE EXPONENT MODIFIER. 8

A relocatable expression has been used to specify the exponent modifier.

ASMG015 INVALID LITERAL USAGE. 8

A valid literal is used illegally, e.g., it specifies a receiving field or a register, or it is a Q-Type constant.

ASMG016 INVALID NAME. 8

A name entry is incorrectly specified, e.g., it contains more than 8 characters, it does not begin with a letter, or has a special character imbedded. If the statement is OPSYN the name is not an ordinary symbol or is an assembler operation mnemonic.

MESSAGE NUMBER		SEVERITY CODE
ASMG017	DATA ITEM TOO LARGE.	8

The constant is too large for the data type or for the explicit length; operand field for packed DC exceeds 32 characters and for zoned DC exceeds 16 characters (excluding decimal points).

ASMG018	INVALID SYMBOL.	8

The symbol is specified invalidly, e.g., it is longer than 8 characters. If the statement is OPSYN the operand entry is not an ordinary symbol or is an assembler operation mnemonic.

ASMG019	EXTERNAL SYMBOL ERROR.	8

One of the following --

1. A symbol appears in the name field of both a CSECT or a COM and a DSECT statement.
2. A symbol appearing in the name field of a DXD instruction also appears in the name field of another DXD instruction, in the operand field of an EXTRN or WXTRN instruction, or in the name field of a CSECT, COM, or DSECT statement.
3. A symbol appearing in the operand field of an EXTRN or WXTRN instruction also appears in the operand field of the same or another EXTRN or WXTRN instruction, or in the name field of a DXD, CSECT, COM, or DSECT instruction.
4. A symbol previously encountered in the name field of a statement other than those mentioned above, appears in the operand field of an EXTRN or WXTRN instruction or in the name field of DXD, CSECT, COM, or DSECT instruction.

ASMG020	INVALID IMMEDIATE FIELD.	8

The value of the immediate operand exceeds 255, (or 9 for SRP) or the operand is not an acceptable type.

ASMG021	SYMBOL NOT PREVIOUSLY DEFINED.	8

An expression requiring that all symbols be previously defined contains at least one symbol not previously defined.

ASMG022	ESD TABLE OVERFLOW.	12

The combined number of control sections and dummy sections plus the number of unique symbols in EXTRN and WXTRN statements and V-type constants exceeds 255. (A DSECT which appears as XD makes two entries).

MESSAGE NUMBER		SEVERITY CODE
ASMG023	PREVIOUSLY DEFINED NAME.	8
	The symbol which appears in the name field has appeared in the name field of a previous statement.	
ASMG024	UNDEFINED SYMBOL.	8
	A symbol being referenced has not been defined in the program.	
ASMG025	RELOCATABILITY ERROR.	8
	A relocatable or complex relocatable expression is specified where an absolute expression is required, an absolute expression or complex relocatable expression is specified where a relocatable expression is required, or a relocatable term is involved in multiplication or division.	
ASMG026	TOO MANY LEVELS OF PARENTHESES.	12
	An expression specifies more than 11 levels, 5 levels if NOEXTEN, of parentheses.	
ASMG027	TOO MANY TERMS.	12
	More than 25 terms, 16 terms if NOEXTEN, are specified in an expression.	
ASMG028	REGISTER NOT USED.	4
	A register specified in a DROP statement is not currently in use.	
ASMG029	CCW ERROR.	8
	Bits 37-39 of the CCW are set to non-zero.	
ASMG030	INVALID CNOP.	12
	An invalid combination of operands is specified in a CNOP instruction.	
ASMG031	UNKNOWN TYPE.	8
	Incorrect type designation is specified in a DC, DS, or literal. If the DOS option was specified, then L and Q types will be flagged.	

MESSAGE NUMBER — SEVERITY CODE

ASMG032 OP-CODE NOT ALLOWED TO BE GENERATED. — 8

An operation code allowed only in source statements has been obtained through substitution of a value for a variable symbol.

ASMG033 ALIGNMENT ERROR. — 4

Referenced address is not aligned to the proper boundary for this instruction, e.g., start operand not a multiple of 8. This message is not produced if a base or index register is explicitly specified in the operand. If PARM=NOALGN was specified, then it is only produced for BC, BXH, BXLE, BAL, BCT, EX, and LPSW.

ASMG034 INVALID OP-CODE. — 8

Syntax error, e.g., more than 8 characters in operation field, not followed by blank on first card, op code missing.

ASMG035 ADDRESSABILITY ERROR. — 8

The referenced address does not fall within the range of a USING instruction.

ASMG037 MNOTE STATEMENT. — variable

This indicates that an MNOTE statement has been generated from a macro definition. The text and severity code of the MNOTE statement will be found in line in the listing.

ASMG038 ENTRY ERROR. — 8

A symbol in the operand of an ENTRY statement appears in more than one ENTRY statement, it is undefined, it is defined in a dummy section or in blank common, or it is equated to a symbol defined by an EXTRN or WXTRN statement.

ASMG039 INVALID DELIMITER. — 12

This message can be caused by any syntax error, e.g., missing delimiter, special character used which is not a valid delimiter, delimiter used illegally, operand missing, i.e., nothing between delimiters, unpaired parentheses, imbedded blank in expression.

MESSAGE NUMBER		SEVERITY CODE
ASMG040	GENERATED RECORD TOO LONG. There are more than 236 characters in a generated statement.	12
ASMG041	UNDECLARED VARIABLE SYMBOL. Variable symbol is not declared in a define SET symbol statement or in a macro prototype.	8
ASMG042	SINGLE TERM LOGICAL EXPRESSION IS NOT A SETB SYMBOL. The single term logical expression has not been declared as a SETB symbol.	8
ASMG043	SET SYMBOL PREVIOUSLY DEFINED. Self-explanatory.	8
ASMG044	SET SYMBOL USAGE INCONSISTENT WITH DECLARATION. A SET symbol has been declared as undimensioned, but is subscripted, or has been declared dimensioned, but is unsubscripted.	8
ASMG045	ILLEGAL SYMBOLIC PARAMETER. An attribute has been requested for a variable symbol which is not a legal symbolic parameter.	8
ASMG046	AT LEAST ONE RELOCATABLE Y TYPE CONSTANT IN ASSEMBLY. One or more relocatable Y type constants in assembly; relocation may result in address greater than 2 bytes in length. This diagnostic cannot occur if NOYFLAG option is specified.	4
ASMG047	SEQUENCE SYMBOL PREVIOUSLY DEFINED. Self-explanatory.	12
ASMG048	SYMBOLIC PARAMETER PREVIOUSLY DEFINED OR SYSTEM VARIABLE SYMBOL DECLARED AS SYMBOLIC PARAMETER. Self-explanatory.	12
ASMG049	VARIABLE SYMBOL MATCHES A PARAMETER. Self-explanatory.	12

MESSAGE NUMBER		SEVERITY CODE
ASMG050	INCONSISTENT GLOBAL DECLARATIONS.	8

A global SET variable symbol, defined in more than one macro definition or defined in a macro definition and in the source program, is inconsistent in SET type or dimension. In the case of GBLC, it may be the length definitions which are inconsistent.

ASMG051 MACRO DEFINITION PREVIOUSLY DEFINED. 12

Prototype operation field is the same as a machine or assembler instruction or a previous prototype. This message is not produced when a programmer macro matches a library macro. The programmer macro will be assembled with no indication of the corresponding library macro.

ASMG052 NAME FIELD CONTAINS ILLEGAL SET SYMBOL. 8

SET symbol in name field does not correspond to SET statement type.

ASMG055 INVALID EXECUTE CARD PARAMETER(S). 8

Self-explanatory. This message will erroneously reference the 1st or 2nd statement in the program. It is printed whenever ASMG255 is printed.

ASMG056 ARITHMETIC OVERFLOW. 8

The intermediate or final result of an expression is not within the range of -2**31 to 2**31-1.

ASMG057 SUBSCRIPT NOT WITHIN DIMENSION. 8

&SYSLIST or symbolic parameter subscript exceeds 200, or is less than one, or set symbol subscript exceeds dimension specified in LCL or GBL statement.

ASMG058 RE-ENTRANCY VIOLATION. 4

This instruction has been flagged because, when executed, it may store data into a control section or a common area. This message is generated only when requested via PARM=RENT and merely indicates a possible re-entrant error.

ASMG059 UNDEFINED SEQUENCE SYMBOL. 12

Self-explanatory.

ASMG DIAGNOSTIC MESSAGES

MESSAGE NUMBER		SEVERITY CODE
ASMG060	ILLEGAL ATTRIBUTE NOTATION.	8
	L', S', or I' requested for a parameter whose type attribute does not allow these attributes to be requested.	
ASMG061	ACTR COUNTER EXCEEDED.	12
	Conditional assembly loop counter exceeded; conditional assembly terminated. ASMG divides the ACTR by 2 each time an error is detected during macro expansion.	
ASMG062	GENERATED STRING GREATER THAN 255 CHARACTERS.	8
	Self-explanatory.	
ASMG063	EXPRESSION 1 OF SUBSTRING IS ZERO OR MINUS.	8
	Self-explanatory.	
ASMG064	EXPRESSION 2 OF SUBSTRING IS ZERO OR MINUS.	8
	Self-explanatory.	
ASMG065	INVALID OR ILLEGAL TERM IN ARITHMETIC EXPRESSION.	8
	The value of a SETC symbol used in the arithmetic expression is not composed of decimal digits, or the parameter is not a self-defining term. If PARM=EXTEN, then the value of a SETC symbol used in an arithmetic expression was not a properly formed self-defining term.	
ASMG066	UNDEFINED OR DUPLICATE KEYWORD OPERAND.	12
	The same keyword operand occurs more than once in the macro instruction; a keyword is not defined in a prototype statement; in a mixed mode macro instruction, more positional operands are specified than are specified in the prototype.	
ASMG067	EXPRESSION 1 OF SUBSTRING GREATER THAN LENGTH OF CHARACTER EXPRESSION.	8
	Self-Explanatory.	
ASMG068	ILLEGAL LENGTH SPECIFICATION IN GBL OR LCL STATEMENT.	8
	The length specified in a GBLC or LCLC statement is other than 1 to 255.	

MESSAGE NUMBER		SEVERITY CODE
ASMG069	VALUE OF EXPRESSION 2 OF SUBSTRING TOO LARGE.	8
	If PARM=NOEXTEN, then the value of expression 2 of the substring notation was greater than 8. If PARM=EXTEN, then the value of expression 2 of the substring notation was greater than 255.	
ASMG070	FLOATING POINT CHARACTERISTIC OUT OF RANGE.	12
	Exponent too large for length of defining field, exponent modifier has caused loss of all significant digits.	
ASMG071	ILLEGAL OCCURRENCE OF LCL, GBL, OR ACTR STATEMENT	8
	LCL, GBL, or ACTR statement is not in proper place in the program. This diagnostic cannot occur under EXTEN.	
ASMG072	ILLEGAL RANGE ON ISEQ STATEMENT.	4
	One or more columns to be sequence checked are between the 'begin' and the 'end' columns of the statement.	
ASMG073	ILLEGAL NAME FIELD.	8
	Either a statement requires a name and the name field is blank or a statement has a name which should be blank.	
ASMG074	ILLEGAL STATEMENT IN COPY CODE OR SYSTEM MACRO.	8
	A statement brought in by a COPY statement is END, ICTL, ISEQ, MACRO, MEND, OPSYN or COPY. Under the EXTEN option, MACRO and MEND are valid if not already within a Macro definition, COPY is valid within COPY up to five nesting levels and END is valid if not within a Macro Definition. A model statement in a library macro definition is END, ICTL, ISEQ, OPSYN or PRINT (PRINT is OK if EXTEN option).	
ASMG075	ILLEGAL STATEMENT OUTSIDE OF A MACRO DEFINITION.	8
	Statement allowed only in a macro definition encountered in open code, e.g., .* comment or MNOTE statement.	
ASMG076	SEQUENCE ERROR.	12
	Sequence error discovered in the input stream by the sequence checking mechanism initiated by an ISEQ instruction.	

MESSAGE NUMBER		SEVERITY CODE
ASMG077	ILLEGAL CONTINUATION CARD.	8

Either there are too many continuation cards, or there are non-blanks between the begin and continue columns on the continuation card, or a card not intended as continuation was treated as such because of punch in continue column of preceeding card.

ASMG078	FOLLOWING ERRORS OCCURED WHILE EDITING LIBRARY MACROS.	0

Any error messages which follow this one were generated while processing the library macros used by the program. Comment cards are generated following the END statement telling which library macros had the errors if FULLLIST was not specified. The statement(s) in error follow as generated statements without columns 73-80. Recommended action to determine the statement which is in error --

1) Place the erroneous macro definitions in front of the program as programmer macros, or
2) Concatenate the erroneous SYSLIB members on the front of SYSIN, making them look like programmer macros, or
3) Under EXTEN, COPY the MACRO definition in front of the open code program, or
4) Specify FULLLIST in the EXEC card PARM field.

ASMG079	ILLEGAL STATEMENT IN MACRO DEFINITION.	8

This operation is not allowed within a macro definition.

ASMG080	ILLEGAL START CARD.	8

Statements affecting or depending upon the location counter have been encountered before a START statement.

ASMG081	ILLEGAL FORMAT IN GBL OR LCL STATEMENTS.	8

An operand is not a variable symbol.

ASMG082	ILLEGAL DIMENSION SPECIFICATION IN GBL OR LCL STATEMENT.	8

Dimension is other than 1 to 2500; 1 to 9999 if EXTEN.

ASMG083	SET STATEMENT NAME FIELD NOT A VARIABLE SYMBOL.	8

Self-explanatory.

MESSAGE NUMBER — SEVERITY CODE

ASMG084 ILLEGAL OPERAND FIELD FORMAT. — 8

Syntax invalid, e.g., AIF statement operand does not start with a left parenthesis; operand of AGO is not a sequence symbol; operand of PUNCH, TITLE, MNOTE not enclosed in quotes.

ASMG085 INVALID SYNTAX IN EXPRESSION. — 3

Invalid delimiter, too many terms in expression, too many levels of parentheses, two operators in succession, two terms in succession, or illegal character.

ASMG086 ILLEGAL USAGE OF SYSTEM VARIABLE SYMBOL. — 8

A system variable symbol appears in the name field of a set statement, is declared in a GBL or LCL statement, or is an unsubscripted &SYSLIST in a context other than N'&SYSLIST.

ASMG087 NO ENDING APOSTROPHE. — 8

There is an unpaired apostrophe or ampersand in the statement.

ASMG088 UNDEFINED OPERATION CODE. — 12

Symbol in operation code field does not correspond to a valid machine or assembler operation code or to any operation code in a macro prototype statement. If the statement is OPSYN, the operand entry is not a defined machine or extended operation code, or the operand entry is omitted and the name entry is not a defined machine or extended operation code. May be due to incorrect INSTSET= parameter on EXEC card.

ASMG089 INVALID ATTRIBUTE NOTATION. — 8

Syntax error inside a macro definition, e.g., the argument of the attribute reference is not a symbolic parameter.

ASMG090 INVALID SUBSCRIPT. — 8

Syntax error, e.g., double subscript where single subscript is required or vice versa; not right parenthesis after subscript.

MESSAGE NUMBER		SEVERITY CODE
ASMG091	INVALID SELF-DEFINING TERM.	8
	Value is too large or is inconsistent with the data type, e.g., severity code of MNOTE statement greater than 255.	
ASMG092	INVALID FORMAT FOR VARIABLE SYMBOL.	8
	The first character after the ampersand is not alphabetic, or the variable symbol contains more than 8 characters, or failure to use double ampersand in TITLE card or character self-defining term.	
ASMG093	UNBALANCED PARENTHESIS OR EXCESSIVE LEFT PARENTHESES.	8
	End of statement or card encountered before all parenthesis levels are satisfied. May be caused by embedded blank or other unexpected terminator, or failure to have a punch in continuation column.	
ASMG094	INVALID OR ILLEGAL NAME OR OPERATION IN PROTOTYPE STATEMENT.	12
	Name not blank or variable symbol, or variable symbol in name field is subscripted, or violation of rules for forming a variable symbol, or statement following 'MACRO' is not a valid prototype statement.	
ASMG095	ENTRY TABLE OVERFLOW.	8
	Number of ENTRY symbols, I.E., ENTRY instruction operands, exceeds 100.	
ASMG096	MACRO INSTRUCTION OR PROTOTYPE OPERAND EXCEEDS 255 CHARACTERS IN LENGTH.	12
	Self-explanatory.	

MESSAGE NUMBER		SEVERITY CODE

ASMG097 INVALID FORMAT IN MACRO INSTRUCTION OPERAND OR PROTOTYPE PARAMETER. — 12

This message can be caused by --

1. Illegal '='.
2. A single '&' appears somewhere in the standard value assigned to a prototype keyword parameter.
3. First character of a prototype parameter is not '&'.
4. Prototype parameter is a subscripted variable symbol.
5. Invalid use of alternate format in prototype statement, e.g.,

```
     10      16                              72
     PROTO   &A,&B,
          or
     PROTO   &A,&B,&C                        X
```

6. Unintelligible prototype parameter, e.g., '&A*' or '&A&&'.
7. Illegal (non-assembler) character appears in prototype parameter or macro-instruction operand.

ASMG098 EXCESSIVE NUMBER OF OPERANDS OR PARAMETERS. — 12

Either the prototype has more than 200 parameters or the macro instruction has more than 200 operands.

ASMG099 POSITIONAL MACRO INSTRUCTION OPERAND, PROTOTYPE PARAMETER OR EXTRA COMMA FOLLOWS KEYWORD. — 12

Self-explanatory. This diagnostic cannot occur under EXTEN.

ASMG100 STATEMENT COMPLEXITY EXCEEDED. — 8

More than 32 operands in a DC, DS, DXD, or literal DC, or more than 50 terms in a statement.

ASMG101 EOD ON SYSIN. — 12

End of data encountered in input stream before END card.

ASMG102 INVALID OR ILLEGAL ICTL. — 16

The operands of the ICTL are out of range, or the ICTL is not the first statement in the input deck. The assembly is terminated and further input is ignored.

ASMG103 ILLEGAL NAME IN OPERAND FIELD OF COPY CARD. — 12

Syntax error, e.g., symbol has more than 8 characters or has an illegal character.

MESSAGE NUMBER		SEVERITY CODE
ASMG104	COPY CODE NOT FOUND.	12
	The operand of a copy statement specified copy text which cannot be found in the library.	
ASMG105	EOD ON SYSTEM MACRO LIBRARY.	12
	End of data encountered on library member before MEND card.	
ASMG106	NOT NAME OF DSECT OR DXD.	8
	Referenced symbol expected to be DSECT name, but it is not.	
ASMG107	INVALID OPERAND.	8
	Invalid syntax in DC operand, e.g., invalid hexadecimal character in hexadecimal DC; operand string too long for X, B, C, DC's; operand unrecognizable, contains invalid value, or incorrectly specified.	
ASMG108	INVALID EQU ARGUMENT.	8
	Under EXTEN the second operand defines the length of the symbol in the name field and operand three defines its type. Operand two is not in the range 0-65535 or operand is not a self-defining term in the range 0-255.	
ASMG109	PRECISION LOST.	8
	Self-explanatory.	
ASMG110	EXPRESSION VALUE TOO LARGE.	8
	Value of expression greater than -16777216 to +1677215. Expressions in EQU and ORG statements are flagged if (1) they include terms previously defined as negative values, or (2) positive terms give a result of more than three bytes in magnitude. The error indication may be erroneous due to (1) the treatment of negative values as three-byte positive values, or (2) the effect of large positive values on the location counter if a control section begins with a START statement having an operand greater than zero, or a control section is divided into subsections.	

MESSAGE NUMBER		SEVERITY CODE
ASMG111	INVALID PRINT, PUSH OR POP OPERAND.	8
	The operands of PRINT were not ON, OFF, GEN, NOGEN, DATA, NODATA or the PRINT operands conflict with one another. The operands of PUSH/POP were not PRINT or USING. Duplicate PUSH/POP operands are valid as each occurrence bumps the push down stack by one level.	
ASMG112	INVALID PUSH/POP REQUEST.	8
	A PUSH request requires more than 5 levels of stacking for a PRINT or USING request. A POP request for PRINT or USING was not preceeded by a corresponding PUSH.	
ASMG114	INSUFFICIENT MEMORY FOR USING MAP.	4
	The UMAP option requires sixteen bytes of storage per register specified in a USING statement plus sixteen bytes per register reinstated with a POP USING statement.	
ASMG115	UPDATE CONDITION CODE EXCEEDED.	variable
	The UPCOND= keyword parameter sets a condition code that the UPDATE option tests. If the UPCOND= value is exceeded the assembly, or all subsequent assemblies if BATCH, will terminate at the start of macro expansion phase. The severity code is the UPDATE code that exceeded UPCOND.	
ASMG116	ILLEGAL OPSYN.	8
	An OPSYN statement may be preceeded only by an ICTL instruction or another OPSYN statement.	
ASMG200	UNABLE TO OPEN (DDNAME). CHECK CONTROL CARD.	
	The assembler could not open one of the files. Check that the DD card is present. The message is typed for SYSPRINT, unless SYSTERM is open, and is printed for others. For SYSPUNCH, SYSLIN, SYSTERM or SYSUP ASMG204 is also printed.	
	If SYSPUNCH, then NODECK option is assumed.	4
	If SYSTERM, then NOTERM is assumed.	4
	If SYSUP, then NOUPDATE is assumed.	16
	If SYSLIN (SYSGO), then NOLOAD option is assumed.	16
	For all others assembly is terminated.	20

MESSAGE NUMBER		SEVERITY CODE
ASMG201	ILLEGAL BLKSIZE ON (DDNAME). Either the DD card or the data set label had a blksize that was not allowed. The assembler ignores the invalid blocksize. Message ASMG203 is also printed.	4
ASMG202	UNPROCESSED (SYSIN DDNAME) AND/OR (SYSUP DDNAME) RECORDS EXIST. Under the UPDATE option there were records in the SYSIN data set and/or the SYSUP data set that were not read by either the assembler or the user's program if EXECUTE. If the update log is not present check SYSUP for incorrect sequencing. This message cannot occur in BATCH mode.	0
ASMG203	ILLEGAL DCB OPERANDS. SEE FIRST PAGE. One or more ASMG201 messages were printed at the start of the assembly.	4
ASMG204	UNOPENABLE DATA SETS. SEE FIRST PAGE. One or more ASMG200 messages were printed at start of assembly. Severity code depends on which data sets could not be opened. See ASMG200.	4 or 16
ASMG205	UNPROCCESSED (SYSIN DDNAME) RECORDS EXIST. There were cards in the SYSIN data set that were not read by either the assembler or the user's program if executed. This message cannot occur in BATCH mode.	0
ASMG206	NNNNN I/O ERRORS ON (SYSPRINT DDNAME). The operating system entered the SYSPRINT SYNAD routine NNNNN times.	4
ASMG207	NNNNN I/O ERRORS ON (SYSPUNCH DDNAME). The operating system entered the SYSPUNCH SYNAD routine NNNNN times.	4
ASMG208	MORE THAN NNNNN MACROS IN LIBRARY. There were too many macros in the SYSLIB data sets. No action is required unless ASMG209 follows immediately. NNNNN = 1000 is the default.	0

MESSAGE NUMBER		SEVERITY CODE
ASMG209	NNNNN LIBRARY FINDS DONE TO COMPLETE ASSEMBLY.	0

This message is a performance diagnostic. The CS FIND routine was used NNNNN times to search for a macro not in the incore macro table.

ASMG255	ERROR IN PARM FIELD.	8

An unrecognizable option name or a numeric quantity out of range was found in the PARM field on the EXEC card. Processing of the PARM field is terminated when the first error is encountered. Message ASMG055 is also printed.

ASMG300	FEATURES INCOMPATIBLE WITH EXECUTE CPTION USED.	0

Use was made of one of the following unsupported features -- DXD'S, CXD'S, Q-TYPE constants, or named DSECTS. Program loading is terminated immediately.

ASMG302	INSUFFICIENT MEMORY TO LOAD ABOVE CSECT.	20

The combined memory requirements of those CSECTS whose names have been printed in the load map exceed the amount of available memory. Either make more memory available or reduce the size of the executed program or decrease the blksize of some QSAM data sets.

ASMG303	UNRESOLVED EXTERNAL REFERENCE. (NAME)	0

The symbol printed was named in an EXTRN statement or a V-TYPE address constant and there is no corresponding entry statement in the program. The printing of load map ceases, but ESD processing is continued to determine if there are other unresolved external references. Execution is inhibited.

ASMG304	EXECUTION ERROR.	0

A program interrupt occured during execution of the user's program. This message is followed by a dump of the PSW, general and floating registers, and the user's memory.

ASMG305	EXCESSIVE EXECUTION TIME. TIME ALLOWED WAS nnnn.nnn SEC.	0

Execution is terminated. The time allowed figure comes from the EXTIME= parameter. A dump as in ASMG304 is produced.

MESSAGE NUMBER		SEVERITY CODE
ASMG306	INSUFFICIENT MEMORY FOR USE OF EXECUTE OPTION WITH ANY PROGRAM. Phase FEX was unable to obtain 6000 bytes for its initial work area. Either increase available memory or decrease blksize of some QSAM data sets.	20
ASMG307	OBJECT FILE MISSING OR INCOMPLETE. EXECUTION DELETED. End-of-file was found on the object deck utility before the object deck end card.	0
ASMG308	END OF FILE ON INPUT DCB. JOB TERMINATED. User did not alter the SYSIN DCB EODAD exit but attempted to read the /* or $JOB card from SYSIN.	0
ASMG320	BLANK (SYSIN DDNAME) SEQUENCE FIELD. A record with a blank sequence number was found on the SYSIN data set. Under UPDATE all SYSIN records must be sequenced. The record is ignored.	12
ASMG321	BLANK (SYSUP DDNAME) SEQUENCE FIELD. A record with a blank sequence number was found on the SYSUP data set. All records, except ./ control cards must be sequenced. The record is ignored.	12
ASMG323	INSERTION. A record from SYSUP is being inserted in the SYSIN card stream. This message is not listed under NOUPLIST.	0
ASMG324	TO BE REPLACED. This SYSIN record is being replaced by a SYSUP record with the same sequence number. This message is not listed under NOUPLIST.	0
ASMG325	REPLACEMENT. This SYSUP record is replacing the SYSIN record listed in the preceeding message. This message is not listed under NOUPLIST.	0
ASMG326	DELETION. This record on SYSIN is being deleted because of a ./ DELETE card. This message is not listed under NOUPLIST.	0

MESSAGE NUMBER		SEVERITY CODE
ASMG327	NO RECORDS IN RANGE.	16
	This ./ DELETE command shown had no effect on the SYSIN data set.	
ASMG328	XXXXXXXX TO XXXXXXXX MISMATCH.	16
	The range of a ./ DELETE card did not exactly match the range deleted on SYSIN. The range actually deleted is shown in the message.	
ASMG329	XXXXXXX RECORDS DELETED.	0
	This message tells how many records were deleted from SYSIN by a ./ DELETE card on SYSUP. This message is not listed if NOUPLIST.	
ASMG330	FLUSHING.	4
	The same error has occurred two or more times in a row. This message indicates a file is being flushed until processing can resume.	
ASMG331	(SYSUP DDNAME) SEQUENCE ERROR.	12
	A sequence error has been detected on output and SYSUP is blamed. This record is not passed to the assembler.	
ASMG332	(SYSIN DDNAME) SEQUENCE ERROR.	12
	Records have been found to be out of sequence upon input from SYSIN. This record is ignored.	
ASMG334	INVALID DELETE OPERANDS.	16
	A ./ DELETE card has been found with improper parameters. Both SEQ1= and SEQ2= must be specified. This command is ignored.	
ASMG335	SEQ1 IS GREATER THAN SEQ2.	16
	A ./ DELETE card with improper sequence numbers as parameters. This command is ignored.	
ASMG336	CONTROL CARD NOT SUPPORTED.	8
	A ./ control card other than DELETE was found on SYSUP. This command will be ignored. This message is not listed if NOUPLIST.	

MESSAGE NUMBER		SEVERITY CODE
ASMG337	CONTROL CARD NOT RECOGNIZED.	16
	A ./ control card has been found with a command that is not recognized by IEBUPDTE. This record will be ignored.	
ASMG338	CONTINUED CONTROL CARD.	0
	This card is taken to be the continuation of the previous ./ control card. This message is not listed if NOUPLIST.	
ASMG339	./ENDUP CARD ON (SYSUP DDNAME).	4
	A ./ ENDUP card has been found on SYSUP and it is treated as the end of file marker. If a ./ ENDUP is absent the real end of file mark terminates SYSUP processing.	
ASMG340	UNPROCESSED (SYSUP DDNAME) RECORDS.	12
	There were cards in the SYSUP data set following a ./ ENDUP card. These cards have been ignored. The record accompanying this message is the first record after the ./ ENDUP.	
ASMG341	none	0
	This SYSIN record is being passed to the assembler by the UPDATE option. Listed only under FULLUPLIST.	
ASMG342	SYSUP NUMBERING RECORD.	4
	A ./ NUMBER card has been processed in SYSUP.	
ASMG343	COMMENT CONTROL CARD.	4
	A ./ COMMENT card has been processed in SYSUP. This is an extension to the standard IEBUPDTE function list.	
ASMG501A	EDIT-FORMAT RECORD INVALID OR > 80 CHARS, DDNAME = XXXXXXXX.	20
	A WYLBUR format input file contains invalid data that cannot be processed by the Assembler. This diagnostic can only occur with an initial program name of ASMGWYL.	

MESSAGE NUMBER		SEVERITY CODE
ASMG502A	V-FORMAT INPUT, DDNAME = XXXXXXXX.	20

An input file is being processed with RECFM other than 'F' or 'U' whan the initial program name of ASMGWYL was specified.

ASMG503A	BLOCK EXCEEDS DECLARED BLKSIZE, DDNAME = XXXXXXXX.	20

A WYLBUR format input file is being processed and the buffer space required to unsquish a block is not available. Specify a larger BLKSIZE= for the first file in the concatenation.

ASMG989I	INSUFFICIENT MEMORY FOR PHASE F3 DICTIONARIES.	20

The storage requirements of the generation time dictionaries exceeded available memory. Either increase available memory, decrease data set blksizes, or decrease UTBUFF= parameter.

ASMG990I	INSUFFICIENT MEMORY TO BUFFER UTILITIES.	20

The buffering routine found that it was unable to keep even one record from each buffered utility in memory. Either increase available memory, decrease data set blksizes, or decrease UTBUFF= parameter. This error may also occur instead of ASMG989I in an infinitely recursive macro sequence.

ASMG992I	INSUFFICIENT DICTIONARY SPACE FOR PHASE F2.	20

The global dictionary plus the largest local dictionary required more memory that was available. Either increase available memory, decrease data set blksizes, or decrease UTBUFF= parameter.

ASMG993I	INSUFFICIENT MEMORY FOR PHASE F2 I/O BUFFERS.	20

There was not enough memory available to allocate I/O buffers. Either increase available memory, decrease data set blksizes, or decrease the UTBUFF= parameter.

ASMG994I	INSUFFICIENT MEMORY TO PROCESS RLD.	20

Sorting the RLD required more memory than was available. Either increase available memory or decrease UTBUFF= parameter.

MESSAGE NUMBER — SEVERITY CODE

ASMG995I INSUFFICIENT MEMORY TO PROCESS XREF. — 20

Building the XREF table required more memory than was available. Either increase available memory or decrease UTBUFF= parameter.

ASMG996I INSUFFICIENT MEMORY TO PROCESS SYMBOL TABLE. — 20

I/O buffers, hash table, LAT/LBT, ESD, and symbol table required more memory than was available. Either increase available memory, decrease data set blksizes, or decrease UTBUFF= parameter. Use of the XREF and LREF options increase the size of the Symbol table.

ASMG998I INSUFFICIENT MEMORY TO SATISFY MINIMUM SPACE REQUIREMENTS. — 20

If SPACE=MAX specified, less than 12736 bytes available.
If SPACE=nnn specified, less than nnn bytes available.
If SPACE=MAX-nnn specified, less than 12736+nnn bytes available.
Reduce QSAM blocking/buffering if possible, or increase partition or region size.

ASMG999A ASSEMBLY TERMINATED. I/O ERROR. SYNADAF INFO='text' — 20

The operating system entered the SYNAD exit of one of the assembler's DCBs other than SYSPRINT or SYSPUNCH. 'text' is the text produced by the SYNADAF macro except that the jobname and stepname are omitted. This message is both typed and printed.

APPENDIX B

ASMG INSTRUCTION SETS

NOTE -- All of the instruction sets have all of the Assembler mnemonics except instruction set 00 which does not have PUSH or POP and instruction set 09 which does not have CXD, DXD, OPSYN, PUSH or POP.

MACHINE NMEMONIC CODES

MNEM CODE	INSTRUCTION	HEX OP	MACH TYPE	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
A	ADD	5A	RX	R1,D2(X2,B2)	*	*		*	*	*	*
AD	ADD NORMALIZED, LONG	6A	RX	R1,D2(X2,B2)	*	*		*	*	*	*
ADR	ADD NORMALIZED, LONG	2A	RR	R1,R2	*	*		*	*	*	*
AE	ADD NORMALIZED, SHORT	7A	RX	R1,D2(X2,B2)	*	*		*	*	*	*
AER	ADD NORMALIZED, SHORT	3A	RR	R1,R2	*	*		*	*	*	*
AXR	ADD NORMALIZED (EXTENDED)	36	RR	R1,R2	*				*		*
AH	ADD HALF WORD	4A	RX	R1,D2(X2,B2)	*	*	*	*	*	*	*
AL	ADD LOGICAL	5E	RX	R1,D2(X2,B2)	*	*		*	*	*	*
ALR	ADD LOGICAL	1E	RR	R1,R2	*	*		*	*	*	*
AP	ADD DECIMAL	FA	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
AR	ADD	1A	RR	R1,R2	*	*	*	*	*	*	*
AU	ADD UNNORMALIZED, SHORT	7E	RX	R1,D2(X2,B2)	*	*		*	*	*	*
AUR	ADD UNNORMALIZED, SHORT	3E	RR	R1,R2	*	*		*	*	*	*
AW	ADD UNNORMALIZED, LONG	6E	RX	R1,D2(X2,B2)	*	*		*	*	*	*
AWR	ADD UNNORMALIZED, LONG	2E	RR	R1,R2	*	*		*	*	*	*
BAL	BRANCH AND LINK	45	RX	R1,D2(X2,B2)	*	*		*	*	*	*
BALR	BRANCH AND LINK	05	RR	R1,R2	*	*		*	*	*	*
BAS	BRANCH AND STORE	4D	RX	R1,D2(X2,B2)			*			*	
BASR	BRANCH AND STORE	0D	RR	R1,R2			*			*	
BC	BRANCH ON CONDITION	47	RX	M1,D2(X2,B2)	*	*	*	*	*	*	*
BCR	BRANCH ON CONDITION	07	RR	M1,R2	*	*	*	*	*	*	*

MNEM CODE	INSTRUCTION	HEX OP	MACH TYPE	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
BCT	BRANCH ON COUNT	46	RX	R1,D2(X2,B2)	*	*		*	*	*	*
BCTR	BRANCH ON COUNT	06	RR	R1,R2	*	*		*	*	*	*
BXH	BRANCH ON INDEX HIGH	86	RS	R1,R3,D2(B2)	*	*		*	*	*	*
BXLE	BRANCH ON INDEX LOW OR EQUAL	87	RS	R1,R3,D2(B2)	*	*		*	*	*	*
C	COMPARE ALGEBRAIC	59	RX	R1,D2(X2,B2)	*	*		*	*	*	*
CD	COMPARE,LONG	69	RX	R1,D2(X2,B2)	*	*		*	*	*	*
CDR	COMPARE,LONG	29	RR	R1,R2	*	*		*	*	*	*
CDS	COMPARE DOUBLE AND SWAP	BB	RS	R1,R3,D2(B2)	*						*
CE	COMPARE,SHORT	79	RX	R1,D2(X2,B2)	*	*		*	*	*	*
CER	COMPARE,SHORT	39	RR	R1,R2	*	*		*	*	*	*
CH	COMPARE HALF WORD	49	RX	R1,D2(X2,B2)	*	*	*	*	*	*	*
CHPM	CHANGE PRIORITY MASK	B3	SI	D1(B1),I2				*			
CIO	CONTROL I/O	98	SI	D1(B1),OF			*				
CL	COMPARE LOGICAL	55	RX	R1,D2(X2,B2)	*	*		*	*	*	*
CLC	COMPARE LOGICAL	D5	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
CLCL	COMPARE LOGICAL LONG	0F	RR	R1,R2	*						*
CLI	COMPARE LOGICAL	95	SI	D1(B1),I2	*	*	*	*	*	*	*
CLM	COMPARE LOGICAL UNDER MASK	BD	RS	R1,M3,D2(B2)	*						*
CLR	COMPARE LOGICAL	15	RR	R1,R2	*	*		*	*	*	*
CLRIO	CLEAR I/O	9D01	S	D2(B2)	*						*
CP	COMPARE DECIMAL	F9	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
CR	COMPARE ALGEBRAIC	19	RR	R1,R2	*	*		*	*	*	*
CS	COMPARE AND SWAP	BA	RS	R1,R3,D2(B2)	*						*
CVB	CONVERT TO BINARY	4F	RX	R1,D2(X2,B2)	*	*			*	*	*
CVD	CONVERT TO DECIMAL	4E	RX	R1,D2(X2,B2)	*	*			*	*	*

MACHINE NMEMONIC CODES

MNEM CODE	INSTRUCTION	HEX OP	MACH TYPE	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
D	DIVIDE	5D	RX	R1,D2(X2,B2)	*	*		*	*	*	*
DD	DIVIDE LONG	6D	RX	R1,D2(X2,B2)	*	*		*	*	*	*
DDR	DIVIDE,LONG	2D	RR	R1,R2	*	*		*	*	*	*
DE	DIVIDE,SHORT	7D	RX	R1,D2(X2,B2)	*	*		*	*	*	*
DER	DIVIDE,SHORT	3D	RR	R1,R2	*	*		*	*	*	*
DP	DIVIDE DECIMAL	FD	SS	D1(L1,B1(,D2(L2,B2)	*	*	*		*	*	*
DR	DIVIDE	1D	RR	R1,R2	*	*		*	*	*	*
ED	EDIT	DE	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
EDMK	EDIT AND MARK	DF	SS	D1(L,B1),D2(B2)	*	*			*	*	*
EX	EXECUTE	44	RX	R1,D2(X2,B2)	*	*		*	*	*	*
HDR	HALVE,LONG	24	RR	R1,R2	*	*		*	*	*	*
HDV	HALT DEVICE	9E01	SI	D1,B1	*	*		*	*	*	*
HER	HALVE,SHORT	34	RR	R1,R2	*	*		*	*	*	*
HIO	HALT I/O	9E	SI	D1(B1)	*	*		*	*	*	*
HPR	HALT AND PROCEED	99	SI	D1(B1)	*	*		*	*	*	*
HVC	HYPERVISOR CALL	83	RS	R1,R3,D2(B2)	INSTSET=71 ONLY						
IC	INSERT CHARACTER	43	RX	R1,D2(X2,B2)	*	*		*	*	*	*
ICM	INSERT CHARACTERS UNDER MASK	BF	RS	R1,M3,D2(B2)	*						*
IPK	INSERT PSW KEY	B20B	S	----	*						*
ISK	INSERT STORAGE KEY	09	RR	R1,R2	*	*		*	*	*	*
L	LOAD	58	RX	R1,D2(X2,B2)	*	*		*	*	*	*
LA	LOAD ADDRESS	41	RX	R1,D2(X2,B2)	*	*		*	*	*	*
LCDR	LOAD COMPLEMENT, LONG	23	RR	R1,R2	*	*		*	*	*	*

MNEM		HEX	MACH		INSTRUCTION SET						
CODE	INSTRUCTION	OP	TYPE	OPERAND FORMAT	00	09	20	44	60	67	70
LCER	LOAD COMPLEMENT, SHORT	33	RR	R1,R2	*	*		*	*	*	*
LCR	LOAD COMPLEMENT	13	RR	R1,R2	*	*		*	*	*	*
LCTL	LOAD CONTROL	B7	RS	R1,R3,D2(B2)	*						*
LD	LOAD,LONG	68	RX	R1,D2(X2,B2)	*	*		*	*	*	*
LDR	LOAD,LONG	28	RR	R1,R2	*	*		*	*	*	*
LE	LOAD,SHORT	78	RX	R1,D2(X2,B2)	*	*		*	*	*	*
LER	LOAD,SHORT	38	RR	R1,R2	*	*		*	*	*	*
LH	LOAD HALF WORD	48	RX	R1,D2(X2,B2)	*	*	*	*	*	*	*
LM	LOAD MULTIPLE	98	RS	R1,R3,D2(B2)	*	*		*	*	*	*
LMC	LOAD MULTIPLE CONTROL	B8	RS	R1,R3,D2(B2)						*	
LNDR	LOAD NEGATIVE, LONG	21	RR	R1,R2	*	*		*	*	*	*
LNER	LOAD NEGATIVE, SHORT	31	RR	R1,R2	*	*		*	*	*	*
LNR	LOAD NEGATIVE	11	RR	R1,R2	*	*		*	*	*	*
LPDR	LOAD POSITIVE, LONG	20	RR	R1,R2	*	*		*	*	*	*
LPER	LOAD POSITIVE, SHORT	30	RR	R1,R2	*	*		*	*	*	*
LPR	LOAD POSITIVE	10	RR	R1,R2	*	*		*	*	*	*
LPSW	LOAD PSW	82	SI	D1(B1)	*	*		*	*	*	*
LPSX	LOAD PSW SPECIAL	B2	SI	D1(B1)				*			
LR	LOAD	18	RR	R1,R2	*	*		*	*	*	*
LRA	LOAD REAL ADDRESS	B1	RX	R1,D2(X2,B2)	*					*	*
LRDR	LOAD ROUNDED (EXTENDED/LONG)	25	RR	R1,R2	*				*		*

MACHINE NMEMONIC CODES

MNEM		HEX	MACH		INSTRUCTION SET						
CODE	INSTRUCTION	OP	TYPE	OPERAND FORMAT	00	09	20	44	60	67	70
LRER	LOAD ROUNDED (LONG/SHORT)	35	RR	R1,R2	*				*		*
LTDR	LOAD AND TEST, LONG	22	RR	R1,R2	*	*		*	*	*	*
LTER	LOAD AND TEST, SHORT	32	RR	R1,R2	*	*		*	*	*	*
LTR	LOAD AND TEST	12	RR	R1,R2	*	*		*	*	*	*
M	MULTIPLY	5C	RX	R1,D2(X2,B2)	*	*		*	*	*	*
MC	MONITOR CALL	AF	SI	D1(B1),I2	*						*
MD	MULTIPLY,LONG	6C	RX	R1,D2(X2,B2)	*	*		*	*	*	*
MDR	MULTIPLY,LONG	2C	RR	R1,R2	*	*		*	*	*	*
ME	MULTIPLY,SHORT	7C	RX	R1,D2(X2,B2)	*	*		*	*	*	*
MER	MULTIPLY,SHORT	3C	RR	R1,R2	*	*		*	*	*	*
MH	MULTIPLY HALF WORD	4C	RX	R1,D2(X2,B2)	*	*		*	*	*	*
MP	MULTIPLY DECIMAL	FC	SS	D1(L1,B1),(L2,B2)	*	*	*		*	*	*
MR	MULTIPLY	1C	RR	R1,R2	*	*		*	*	*	*
MVC	MOVE CHARACTER	D2	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
MVCL	MOVE LONG	0E	RR	R1,R2	*						*
MVI	MOVE IMMEDIATE	92	SI	D1(B1),I2	*	*	*	*	*	*	*
MVN	MOVE NUMERICS	D1	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
MVO	MOVE WITH OFFSET	F1	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
MVZ	MOVE ZONES	D3	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
MXD	MULTIPLY (LONG/EXTENDED)	67	RX	R1,D2(X2,B2)	*				*		*
MXDR	MULTIPLY (LONG/EXTENDED)	27	RR	R1,R2	*				*		*

MNEM		HEX	MACH		INSTRUCTION SET						
CODE	INSTRUCTION	OP	TYPE	OPERAND FORMAT	00	09	20	44	60	67	70
MXR	MULTIPLY (EXTENDED)	26	RR	R1,R2	*				*		*
N	AND LOGICAL	54	RX	R1,D2(X2,B2)	*	*		*	*	*	*
NC	AND LOGICAL	D4	SS	D1(L,B1),D2(B2)	*	*			*	*	*
NI	AND LOGICAL IMMEDIATE	94	SI	D1(B1),I2	*	*	*	*	*	*	*
NR	AND LOGICAL	14	RR	R1,R2	*	*		*	*	*	*
O	OR LOGICAL	56	RX	R1,D2(X2,B2)	*	*		*	*	*	*
OC	OR LOGICAL	D6	SS	D1(L,B1),D2(B2)	*	*			*	*	*
OI	OR LOGICAL IMMEDIATE	96	SI	D1(B1),I2	*	*	*	*	*	*	*
OR	OR LOGICAL	16	RR	R1,R2	*	*		*	*	*	*
PACK	PACK	F2	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
PTLB	PURGE TLB	B20D	S	----	*						*
RDD	READ DIRECT	85	SI	D1(B1),I2	*	*			*	*	*
RDDW	READ DIRECT WORD	B5	SI	D1(B1),I2				*			
RRB	RESET REFERENCE BIT	B213	S	D2(B2)	*						*
S	SUBTRACT	58	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SCK	SET CLOCK	B204	S	D1(B1)	*						*
SCKC	SET CLOCK COMPARATOR	B206	S	D2(B2)	*						*
SD	SUBTRACT NORMALIZED,LONG	6B	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SDR	SUBTRACT NORMALIZED,LONG	2B	RR	R1,R2	*	*		*	*	*	*
SE	SUBTRACT NORMALIZED,SHORT	7B	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SER	SUBTRACT NORMALIZED,SHORT	3B	RR	R1,R2	*	*		*	*	*	*

MNEM CODE	INSTRUCTION	HEX OP	MACH TYPE	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
SH	SUBTRACT HALF WORD	4B	RX	R1,D2(X2,B2)	*	*	*	*	*	*	*
SIGP	SIGNAL PROCESSOR	AE	RS	R1,R3,D2(B2)	*						*
SIO	START I/O	9C	SI	D1(B1)	*	*		*	*	*	*
SIOF	START I/O FAST RELEASE	9C01	SI	D1(B1)	*				*		*
SL	SUBTRACT LOGICAL	5F	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SLA	SHIFT LEFT SINGLE ALGEBRAIC	8B	RS	R1,D2(B2)	*	*		*	*	*	*
SLDA	SHIFT LEFT DOUBLE ALGEBRAIC	8F	RS	R1,D2(B2)	*	*		*	*	*	*
SLDL	SHIFT LEFT DOUBLE LOGICAL	8D	RS	R1,D2(B2)	*	*		*	*	*	*
SLL	SHIFT LEFT SINGLE LOGICAL	89	RS	R1,D2(B2)	*	*		*	*	*	*
SLR	SUBTRACT LOGICAL	1F	RR	R1,R2	*	*		*	*	*	*
SP	SUBTRACT DECIMAL	FB	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
SPKA	SET PSW KEY FROM ADDRESS	B20A	S	D2(B2)	*						*
SPM	SET PROGRAM MASK	04	RR	R1	*	*		*	*	*	*
SPSW	SET PSW	81	SI	D1(B1)			*				
SPT	SET CPU TIMER	B208	S	D2(B2)	*						*
SPX	SET PREFIX	B210	S	D2(B2)	*						*
SR	SUBTRACT	1B	RR	R1,R2	*	*	*	*	*	*	*
SRA	SHIFT RIGHT SINGLE ALGEBRAIC	8A	RS	R1,D2(B2)	*	*		*	*	*	*
SRDA	SHIFT RIGHT DOUBLE ALGEBRAIC	8E	RS	R1,D2(B2)	*	*		*	*	*	*
SRDL	SHIFT RIGHT DOUBLE LOGICAL	8C	RS	R1,D2(B2)	*	*		*	*	*	*
SRL	SHIFT RIGHT SINGLE LOGICAL	88	RS	R1,D2(B2)	*	*		*	*	*	*
SRP	SHIFT AND ROUND DECIMAL	FO	SS	D1(L1,B1),D2(B2),I2	*						*
SSK	SET SYSTEM KEY	08	RR	R1,R2	*	*		*	*	*	*
SSM	SET SYSTEM MASK	80	SI	D1(B1)	*	*		*	*	*	*

NMEM CODE	INSTRUCTION	HEX OP	MACH TYPE	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
ST	STORE	50	RX	R1,D2(X2,B2)	*	*		*	*	*	*
STAP	STORE CPU ADDRESS	B212	S	D2(B2)	*						*
STC	STORE CHARACTER	42	RX	R1,D2(X2,B2)	*	*		*	*	*	*
STCK	STORE CLOCK	B205	S	D1(B1)	*						*
STCKC	STORE CLOCK COMPARATOR	B207	S	D2(B2)	*						*
STCM	STORE CHARACTERS UNDER MASK	BE	RS	R1,M3,D2(B2)	*						*
STCTL	STORE CONTROL	B6	RS	R1,M3,D2(B2)	*						*
STD	STORE LONG	60	RX	R1,D2(X2,B2)	*	*		*	*	*	*
STE	STORE SHORT	70	RX	R1,D2(X2,B2)	*	*		*	*	*	*
STH	STORE HALF WORD	40	RX	R1,D2(X2,B2)	*	*	*	*	*	*	*
STIDC	STORE CHANNEL ID	B203	S	D1(B1)	*						*
STIDP	STORE CPU ID	B202	S	D1(B1)	*						*
STM	STORE MULTIPLE	90	RS	R1,R3,D2(B2)	*	*		*	*	*	*
STMC	STORE MULTIPLE CONTROL	B0	RS	R1,R3,D2(B2)						*	
STNSM	STORE THEN AND SYSTEM MASK	AC	SI	D1(B1),I2	*						*
STOSM	STORE THEN OR SYSTEM MASK	AD	SI	D1(B1),I2	*						*
STPT	STORE CPU TIMER	B209	S	B2(D2)	*						*
STPX	STORE PREFIX	B211	S	D2(B2)	*						*
SU	SUBTRACT UNNORMALIZED,SHORT	7F	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SUR	SUBTRACT UNNORMALIZED,SHORT	3F	RR	R1,R2	*	*		*	*	*	*
SVC	SUPERVISOR CALL	0A	RR	I	*	*		*	*	*	*

MNEM		HEX	MACH		INSTRUCTION SET						
CODE	INSTRUCTION	OP	TYPE	OPERAND FORMAT	00	09	20	44	60	67	70
SW	SUBTRACT UNNORMALIZED,LONG	6F	RX	R1,D2(X2,B2)	*	*		*	*	*	*
SWR	SUBTRACT UNNORMALIZED,LONG	2F	RR	R1,R2	*	*		*	*	*	*
SXR	SUBTRACT NORMAL-IZED (EXTENDED)	37	RR	R1,R2	*				*		*
TCH	TEST CHANNEL	9F00	SI	D1(B1)	*	*		*	*	*	*
TIO	TEST I/O	9D00	SI	D1(B1)	*	*		*	*	*	*
TIOB	TEST I/O AND BRANCH	9A	SI	D1(B1),OF			*				
TM	TEST UNDER MASK	91	SI	D1(B1),I2	*	*	*	*	*	*	*
TR	TRANSLATE	DC	SS	D1(L,B1),D2(B2)	*	*	*		*	*	*
TRT	TRANSLATE AND TEST	DD	SS	D1(L,B1),D2(B2)	*	*	*		*		*
TS	TEST AND SET	93	SI	D1(B1)	*	*		*	*	*	*
UNPK	UNPACK	F3	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*
WRD	WRITE DIRECT	84	SI	D1(B1),I2	*	*			*	*	*
WRDW	WRITE DIRECT WORD	B4	SI	D1(B1),I2				*			
X	EXCLUSIVE OR	57	RX	R1,D2(X2,B2)	*	*		*	*	*	*
XC	EXCLUSIVE OR	D7	SS	D1(L,B1),D2(B2)	*	*			*	*	*
XI	EXCLUSIVE OR IMMEDIATE	97	SI	D1(B1),I2	*	*		*	*	*	*
XIO	EXECUTE I/O	D0	SS	D1(UF,B1),D2(B2)			*				
XR	EXCLUSIVE OR	17	RR	R1,R2	*	*		*	*	*	*
ZAP	ZERO AND ADD	F8	SS	D1(L1,B1),D2(L2,B2)	*	*	*		*	*	*

MNEM		HEX	MACH		OPERAND	INSTRUCTION SET						
CODE	INSTRUCTION	OP	TYPE	MASK	FORMAT	00	09	20	44	60	67	70
B	BRANCH	47	RX	F	D2(X2,B2)	*	*	*	*	*	*	*
	UNCONDITIONAL											
BE	BRANCH EQUAL	47	RX	8	D2(X2,B2)	*	*	*	*	*	*	*
BER	BRANCH EQUAL	07	RR	8	R2			*	*	*	*	*
BH	BRANCH HIGH	47	RX	2	D2(X2,B2)	*	*	*	*	*	*	*
BHR	BRANCH HIGH	07	RR	2	R2			*	*	*	*	*
BL	BRANCH LOW	47	RX	4	D2(X2,B2)	*	*	*	*	*	*	*
BLR	BRANCH LOW	07	RR	4	R2			*	*	*	*	*
BM	BRANCH MINUS,	47	RX	4	D2(X2,B2)	*	*	*	*	*	*	*
	BRANCH MIXED											
BMR	BRANCH MINUS,	07	RR	4	R2			*	*	*	*	*
	BRANCH MIXED											
BNE	BRANCH NOT EQUAL	47	RX	7	D2(X2,B2)	*	*	*	*	*	*	*
BNER	BRANCH NOT EQUAL	07	RR	7	R2			*	*	*	*	*
BNH	BRANCH NOT HIGH	47	RX	D	D2(X2,B2)	*	*	*	*	*	*	*
BNHR	BRANCH NOT HIGH	07	RR	D	R2			*	*	*	*	*
BNL	BRANCH NOT LOW	47	RX	B	D2(X2,B2)	*	*	*	*	*	*	*
BNLR	BRANCH NOT LOW	07	RR	B	R2			*	*	*	*	*
BNM	BRANCH NOT MINUS,	47	RX	B	D2(X2,B2)	*	*	*	*	*	*	*
	BRANCH NOT MIXED											*
BNMR	BRANCH NOT MINUS,	07	RR	B	R2			*	*	*	*	*
	BRANCH NOT MIXED											*
BNO	BRANCH NO OVERFLOW	47	RX	E	D2(X2,B2)	*	*	*	*	*	*	*
	BRANCH NOT ONES											
BNOR	BRANCH NO OVERFLOW	07	RR	E	R2			*	*	*	*	*
	BRANCH NOT ONES											
BNP	BRANCH NOT PLUS	47	RX	D	D2(X2,B2)	*	*	*	*	*	*	*
BNPR	BRANCH NOT PLUS	07	RR	D	R2			*	*	*	*	*

MNEM CODE	INSTRUCTION	HEX OP	MACH TYPE	MASK	OPERAND FORMAT	INSTRUCTION SET 00	09	20	44	60	67	70
BNZ	BRANCH NOT ZERO, BRANCH NOT ZEROS	47	RX	7	D2(X2,B2)	*	*	*	*	*	*	*
BNZR	BRANCH NOT ZERO, BRANCH NOT ZEROS	07	RR	7	R2			*	*	*	*	*
BO	BRANCH OVERFLOW, BRANCH ONES	47	RX	1	D2(X2,B2)	*	*	*	*	*	*	*
BOR	BRANCH OVERFLOW, BRANCH ONES	07	RR	1	R2			*	*	*	*	*
BP	BRANCH PLUS	47	RX	2	D2(X2,B2)	*	*	*	*	*	*	*
BPR	BRANCH PLUS	07	RR	2	R2			*	*	*	*	*
BR	BRANCH UNCONDITIONAL	07	RR	F	R2	*	*	*	*	*	*	*
BZ	BRANCH ZERO, BRANCH ZEROS	47	RX	8	D2(X2,B2)	*	*	*	*	*	*	*
BZR	BRANCH ZERO, BRANCH ZEROS	07	RR	8	R2			*	*	*	*	*
NOP	NO OPERATION	47	RX	0	D2(X2,B2)	*	*	*	*	*	*	*
NOPR	NO OPERATION	07	RR	0	R2	*	*	*	*	*	*	*

```
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ****  ********  ***          ***  ************            **************
  ****  ********  **            **  ************           ***************
  ****  ********  **  *******   **  ************  ******  ****************
  ****  ********  ************  **  *******************  *****************
  ****  ********  ************  **  ******************  ******************
  ****  ********  ***********  ***  *****************  *******************
  ****  ********  *********  *****  *****************  *******************
| *****  ******  ********  *******  *****************  ******AAA**********
| ******  ****  *******  *********  *****************  *****A***A*********
| *******  **  ******  ***********  *****************  *****AAAAA*********
| ********    ******            **            *******  *****A***A*********
| *********  *******            **            *******  *****A***A*********
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  *******     ********     *****  ***  *****  **********    **  **********
  ******       ******       ****  **    ***    ********      *  **********
  *****  ****   ****  *****  ***  *  **  *  **  ******  ****    **********
  ****  ****  *  ***  **********    ****   ****  ****  ******   **********
  **** ****  **  ***  **********   *****  *****  ***  ********  **********
  ********  ***  ****     ******  ******  *****  ***  ********  **********
  *******  ****  ******     ****  ******  *****  ***  ********  **********
  ******  *****  **********  ***  ******  *****  ***  ********  **********
  *****  ******  **********  ***  ******  *****  ***  *******   **********
  ****  *******  **********  ***  ******  *****  ***  ******    **********
  ****  *****    ***  *****  ***  ******  *****  ****  ****  *  **********
  *****       *   ***       ****  ******  *****  *****      **  **********
  ******     ***  ****     *****  ******  *****  ******    ***  **********
  ************************************************************  **********
  ************************************************************  **********
  ************************************************************  **********
  ******************************************************  ****  **********
  *******************************************************  ***  **********
  ********************************************************      **********
  *********************************************************    ***********
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
  ************************************************************************
```