SC28-6479-2

IBM DOS/VS COBOL Compiler and Library Installation Reference Material

Program Product

Program Numbers: 5746-CB1
5746-LM4

IBM

PREFACE

This publication contains the system related information needed to install the IBM DOS/VS COBOL Compiler and its associated COBOL Object-time Subroutine Library. These program products operate under control of the IBM DOS/VS System and the Conversational Monitor System (CMS) component of VM/370 (Virtual Machine Facility/370).

This publication is in two sections, each of which contains information related to a specific system publication. You may insert each section in the related system publication if it makes your use of documentation easier. The related system publications are:

DOS/VS System Generation, Order No. GC33-5377

DOS/VS Messages, Order No. GC33-5379

The first section, "Program Product Installation," is directed to the system programmer or planner responsible for the planning and/or implementation of system generation and maintenance. It contains information corresponding to that found in DOS/VS System Generation. The operating system requirements, minimum machine configuration, work file requirements, and IOCS modules required for the DOS/VS compiler are described, as is the procedure for adding the DOS/VS COBOL Compiler and Subroutine Library to the IBM DOS/VS System. This section also contains storage requirements for the compiler and the library.

The second section, "Diagnostic Messages," is directed to the system operator. It lists all messages issued by the DOS/VS compiler to the operator's console. It is to be used by the operator in conjunction with DOS/VS Messages.

CMS messages produced for DOS/VS COBOL are described in the "Using DOS COBOL Under CMS" section of IBM VM/370 CMS User's Guide for COBOL, Order No. SC28-6469.

ADDITIONAL PUBLICATIONS

Within the text, references are made to the following publications:

DOS/VS System Management Guide, Order No. GC33-5371

DOS/VS COBOL Compiler and Library Programmer's Guide, Order No. SC28-6478

Virtual Machine Facility/370 Planning and System Generation Guide, Order No. GC20-1801

Third Edition (January 1976)

This edition replaces the previous edition (numbered SC28-6479-1) and its technical newsletter (numbered SN20-9120) and makes them both obsolete.

This edition applies to Release 2 of the IBM DOS/VS COBOL Compiler and Library and to all subsequent versions and modifications unless otherwise indicated in new editions or technical newsletters.

Significant changes are summarized under "Summary of Amendments," following the title page. Each technical change is marked by a vertical line to the left of the change.

Information in this publication is subject to significant change. Any such changes will be published in new editions or technical newsletters. Before using the publication, consult the latest IBM System/370 Bibliography, GC20-0001, and the technical newsletters that amend the bibliography, to learn which editions and technical newsletters are applicable and current.

Requests for copies of IBM publications should be made to the IBM branch office that serves you.

Forms for readers' comments are provided at the back of the publication. If the forms have been removed, comments may be addressed to IBM Corporation, General Products Division, Programming Publishing--Department J57, 1501 California Avenue, Palo Alto, California 94304. All comments and suggestions become the property of IBM.

© Copyright International Business Machines Corporation 1973, 1976

Summary of Amendments Number 2

Date of Publication: January 9, 1976
Form of Publication: Revision, SC28-6479-2

CMS Support of DOS/VS COBOL

New: Programming Feature

Release 3 of VM/370 supports DOS/VS COBOL compilaton and execution under CMS. VSAM files are also supported.

Summary of Amendments Number 1

Date of Publication: March 1, 1974
Form of Publication: Revision, SC28-6479-1

Support of New CBL Statement Options

New: Programming Feature

Release 2 of the IBM DOS/VS COBOL Compiler and Subroutine Library supports the programmer productivity aid options VERBSUM/NOVERBSUM, VERBREF/ NOVERBREF, and COUNT/NOCOUNT.

Support of LST Statement (Lister Feature)

New: Programming Feature

Release 2 also supports the lister, a portion of the compiler that processes programs written in American National Standard COBOL to produce a reformatted source code listing with embedded cross-reference information and uniform indenting conventions.

Miscellaneous Changes: Installation Procedures

Maintenance: Documentation Only

Corrections have been made throughout, particularly with reference to the installation procedures themselves.

FIGURES

The IBM DOS/VS COBOL Compiler ("the compiler") and Library is a Program Product that operates under control of the IBM DOS/VS system or the Conversational Monitor System (CMS) component of VM/370. It contains the COBOL compiler, as well as the COBOL subroutines that, when required, are combined by the Linkage Editor with the object program produced by the compiler. It also contains a set of transient routines that can be fetched dynamically into the user's storage area during program execution. And it contains a special set of COBOL statements that can be used to perform 3886 processing. It accepts as input any program written in DOS/VS COBOL.

The DOS/VS COBOL Object-time Subroutine Library ("the library") associated with the DOS/VS compiler is also available as a separate Program Product. It contains COBOL subroutines that, when required, are combined by the Linkage Editor with the object program produced by the DOS/VS compiler. The subroutine library also contains a set of transient routines that can be fetched dynamically into the transient area or loaded dynamically into the user's storage area during program execution.

Link edited programs that have been compiled by the DOS/VS COBOL compiler (5746-CB1) all require the presence of the DOS/VS COBOL Library at execution time. Note that a separate library may not be needed. The library packaged with the compiler can be used if the programs are executing on the same CPU as the one on which they are compiled or if they are executing under CMS; otherwise, there must be a separate library (5746-LM4) for each CPU on which the programs are executing.

The CMS installation must order and install DOS/VS (Release 31 or later). If the DOS/VS COBOL programs access any VSAM data sets, DOS/VS VSAM and DOS/VS Access Method Services must also be installed. Then, the CMS installation must install the DOS/VS COBOL compiler and library. The compiler must be link edited in the background partition of DOS/VS, and all of the compiler's relocatable modules must be link edited using the ACTION REL linkage editor control statement. The compiler and library may reside on the DOS/VS system pack or on private libraries.

The CMS installation must also execute special procedures called EXEC procedures to install the CMS/DOS and CMS/VSAM support on the CMS system. These EXEC procedures are described in the Virtual Machine Facility/370: Planning and System Generation Guide, Order No. GC20-1801.

When the DOS/VS COBOL compiler is executed under CMS, it must be executed in the CMS/DOS environment. Before invoking the DOS/VS COBOL compiler or executing DOS/VS COBOL programs under CMS, be sure that the CMS/DOS environment has been previously generated into your CMS system and is active. Also, when the DOS/VS COBOL compiler and DOS/VS COBOL programs are executed under CMS, the DOS/VS system pack or private libraries must be available to CMS.

This section contains the information you will need to add the IBM DOS/VS COBOL Compiler and its associated COBOL Object-time Subroutine Library to your DOS/VS System.

The first part of this section, "System Requirements," describes the operating system requirements, minimum machine configuration, work file requirements, and LIOCS modules required by the compiler. The second part, "Installation Procedure," gives instructions for adding the compiler and the library to the system. The third part, "Storage Requirements," describes the amount of space required by the compiler and the library.

SYSTEM REQUIREMENTS

For DOS/VS, the compiler and the library operate under control of the IBM Disk Operating System (DOS/VS). A DOS Release 29 system is the minimum level required. The minimum machine configuration and operating system requirements to support the compiler and library are described in this section. Execution time considerations and special requirements for use of the Sort feature and the Symbolic Debug feature are also discussed.

For CMS, the compiler and library are controlled by the CMS/DOS environment. A DOS/VS system pack or DOS/VS private libraries must be available for CMS/DOS use. CMS/DOS requires a DOS/VS Release 31 system as a minimum level. The minimum virtual machine configuration and operating system requirements to support the compiler and library under CMS are described in this section. The Sort feature is not supported under CMS. Other execution-time considerations are also discussed in this section.

OPERATING SYSTEM REQUIREMENTS

If the SORT or MERGE verb is used in the source program under DOS/VS and either the sort work files are on 3400 series tape units or on 3330/3333 DASF, or the SORT-OPTION feature is used, or the ASCII-collating sequence is desired or numeric sort keys are separately signed numeric data items, the DOS/VS SORT/MERGE Program Product, Program Number 5746-SM1, is required. For further information on this product see DOS/VS SORT/MERGE Programmer's Guide, Order No. SC33-4018, and DOS/VS SORT/MERGE Installation Reference Material, Order No. SC33-4026.

During link editing, modules produced by the compiler require subroutines from the library. In addition, subroutines from the library may be required during execution. Therefore, the library must also be part of the system used to control program execution.

CMS requires access to the DOS/VS System pack or the private libraries that contain the DOS/VS COBOL compiler or subroutines. To make the system pack available at virtual address 195 as your C-disk specify the filemode when the CMS/DOS environment is invoked:

```
ACCESS 195      C
SET DOS ON      C
```

To make private libraries at address 193 available, access the disk specifying some CMS mode letter such as D, and issue ASSGN and DLBL commands after the CMS/DOS environment is invoked:

```
ACCESS 193      D
SET DOS ON
ASSGN  SYSCLB   D
DLBL   IJSYSCL  D  DSN ? (SYSCLB)
ASSGN  SYSRLB   D
DLBL   JJSYSKL  D  DSN ? (SYSRLB)
```

MACHINE CONFIGURATION FOR DOS/VS

The minimum DOS/VS machine configuration required to <u>use</u> the DOS/VS COBOL compiler and library is:

1. A System/370 model supported by DOS/VS. At least 60K must be available to the compiler.

2. Six work files.

   The system logical unit SYSLNK must be assigned to a single area (extent) on a 2311, 2314, 2319, 3330, or 3340 mass storage device.

   Five programmer logical units (SYS001 through SYS005) must reside on 2400, 3410, or 3420 tape units, or on 2311, 2314, 2319, 3330, or 3340 mass storage devices. (Two programmer logical units, as well as the operating system, must reside on a mass storage device.)

   If the four remaining logical units reside on tape, there must be a separate tape unit for each data set. If they reside on a mass storage device, there must be enough space on that device. See "Work File Requirements" for further information.

   Workfile assignments must be made as follows:

   ```
   SYS001 -- disk unit
   SYS002 -- disk or tape unit
   SYS003 -- disk or tape unit
   SYS004 -- disk or tape unit
   SYS005 -- disk or tape unit (required when Symbolic Debug is
             requested at compile time)
   SYS006 -- disk unit (required for FIPS)
   ```

   Note that SYSLNK need not be assigned at compile time unless the CATAL or LINK option is in effect.

   The filenames for SYSLNK and SYS001 through SYS006 on the TLBL or DLBL statements are IJSYSLN, IJSYS01, IJSYS02, IJSYS03, IJSYS04, IJSYS05, and IJSYS06, respectively.

3. A device for direct operator communication.

4. A device, such as a card reader, for the job input stream.

5. A device, such as a printer or tape unit, for system output files.

6. The floating-point arithmetic feature, if floating point literals or calculations are used.

Note: All devices currently supported by previous IBM DOS COBOL compilers are supported by the IBM DOS/VS COBOL Compiler. These include: 3504/3505 (with OMR), 3525 (with RCE and combined function processing), 3410/3420 Tapes, 3881 Optical Mark Reader, and 3330 Disk. The DOS/VS compiler also supports the 5425 Multifunction Card Unit, 2560 Multifunction Card Machine, 3203 and 5203 Advanced Printers, 3886 Optical Character Reader, 3340 Disk Facility, 3344 Direct Access Storage, 3350 Direct Access Storage in 3330-1 compatibility mode, and 3540 Diskette Input/Output Unit.

Execution-Time Consideration

The amount of virtual storage must be sufficient to accommodate at least:

- The selected control program.
- Support for the file processing techniques used.
- The load module to be executed.

If symbolic debugging is requested, an additional work file, SYS005, must be assigned at compile time. This file must also be assigned at execution time if symbolic debugging control cards are present, though not necessarily to SYS005.

If execution statistics or the 3886 Optical Character Reader feature are requested, additional virtual storage must be available to accommodate its dynamic space requirements.

Use of the symbolic debugging, flow trace, statement number, execution statistics, option or the 3886 Optical Character Reader feature requires additional storage during execution. See IBM DOS/VS COBOL Compiler and Library Programmer's Guide for further information.

Sort/Merge Feature Considerations

The DOS/VS SORT/MERGE Program Product, Program Number 5746-SM1, must be executed under control of DOS/VS. It requires the following minimum machine configuration:

1. The DOS/VS SORT/MERGE Program Product uses 16K bytes; an additional 6K bytes are needed for DOS and for user-written routines.

   Note: Performance often increases significantly if 50K is available for operation of the Sort/Merge program. At the 100K level, the performance could be even higher.

2. Standard instruction set.

3. At least one 2314, 2319, 3330, 3333, or 3340 work file. (System residence requirements may necessitate having an additional disk storage unit for sorting.)

4. One IBM 1403, 1443, 3203, 5203, or 3211 Printer, or one IBM operator communication device (for example, 3215).

5. One IBM 1442, 2501, 2520, 2540, 3505, 3525, or 2560 Card Reader, or one IBM 2400 or 3400 Series Magnetic Tape Unit (7- or 9-track) assigned to SYSIPT and SYSRDR.

6. Three IBM 2400 or 3400 Series Magnetic Tape Units for work files when tape units are to be used for intermediate storage. For a merge operation, no work units are required.

Symbolic Debug Considerations

To use the Symbolic Debug feature of the DOS/VS compiler, a fifth work file (SYS005) is required at compile time. When symbolic debugging is requested, the compiler records in SYS005 the information needed to produce a symbolic formatted dump if the program terminates abnormally or if dynamic dumps are requested. Thus, the file assigned to SYS005 at compile time must also be assigned at execution time. This file need not necessarily be assigned to SYS005, but it must be assigned in the partition in which the program is executed.

SYS005 can be allocated with the other system work files (SYS001, SYS002, SYS003, SYS004) at system generation time. Each COBOL source program requesting symbolic debugging then uses the same work file. However, unique symbolic debugging work file assignments are required when several COBOL programs are to be compiled in the same job. If there is only one SYS005 allocated for use by all COBOL programs, information recorded on the file during compilation of one program is overlaid by information placed there during compilation of a subsequent program. Then, when the object modules are link edited and executed, the information recorded during compilation will not be available if it has been overlaid.

To avoid the loss of the information needed to produce a symbolic formatted dump, assign a unique symbolic debugging work file for each COBOL source program to be compiled. On a mass storage device, these symbolic debugging work files must be assigned to non-overlapping extents. The following example shows how to use ASSGN statements to define the symbolic debugging work file on disk. Note that the compilation step assigns this data set to SYS005; at execution, it is assigned to SYS006.

```
// JOB SAMPLE
// OPTION SYM,NODECK,NOLISTX,NOXREF,LINK
// ASSGN SYS005,X'192'
// DLBL DEMOXX,99/101,SD
// EXTENT SYS005,111111,1,0,1100,50
// EXEC FCOBOL
      CBL SYMDMP

   (COBOL source deck)

/*
// EXEC LNKEDT
// ASSGN SYS006,X'192'
// DLBL DEMOXX,99/101,SD
// EXTENT SYS006,111111,1,0,1100,50
// EXEC

   (SYMDMP control cards)

/*
/&
```

If the symbolic debugging work file is defined as a tape, one physical reel is required. The file-protect ring must be in the tape reel during compilation and execution.

FIPS Considerations

SYS006 is required only for FIPS processing. After printing the header and any CBL card options on SYSLST, the compiler writes the compilation listing on SYS006 as input to the FIPS phases. The FIPS output is written on SYSLST.

MACHINE CONFIGURATION FOR CMS

The minimum CMS virtual machine configuration will support the DOS/VS COBOL compiler and library.

1. A CMS virtual machine with a minimum of 320K of virtual storage and with the CMS/DOS environment active is required. The compiler uses at least 60K of CMS user storage.

2. System and Logical Units.

   SYSIN/SYSIPT must be assigned to the device that contains the input source file. SYSIN/SYSIPT can be assigned to a reader, tape, or disk.

   The user can issue the ASSGN command for the following units:

   SYSPCH to tape, punch, disk, or IGN
   SYSLST to tape, printer, disk, or IGN
   SYSLOG to terminal
   SYS001-SYS002 to disk
   SYS003-SYS005 to tape or disk
   SYS006 to disk

Note: CMS allows "disk" to also be the 3350 in native mode. CMS does not support the 3540 diskette, 2311 disks, or the 2321 datacell. If SYSIN/SYSIPT is unassigned at compilation time, the CMS/DOS COBOL interface issues an error message and terminates the FCOBOL command.

If SYSPCH, SYSLST, SYSLOG, or SYS0001-SYSnnn is unassigned at compilation time, the CMS/DOS COBOL interface sends the output to the SYSIN disk (if SYSIN is assigned to a read/write disk) or to the user's A-disk.

Note: SYSLNK must not be assigned; it is not supported as a system logical unit under CMS/DOS.

3. Device Support

   The following devices, which are supported by DOS/VS, are not supported by CMS/DOS:

   - Card Readers: 1442, 2560P, 2560S, 2596, 3504, 5425P, and 5425S.

   - Disks: 3540, 2311, 2321.

   - Printers: 2560P, 2560S, 3203, 3525, 5203, 5425P, and 5425S.

   - Other Devices: 3881 OMR, 3886OCR

Not all of the DOS/VS LIOCS modules are supported in CMS/DOS; see the "LIOCS Modules" section of this manual for more information.

Execution-Time Considerations

If symbolic debugging is requested, an additional work file, SYS005, should be assigned at compile time. This file should also be assigned at execution time if symbolic debugging control statements are present, though not necessarily to SYS005. If the file is not assigned, the CMS/DOS COBOL interface directs the output to the SYSIN disk (if SYSIN is assigned to a read/write disk) or to the user's A-disk.

The SORT verb is not supported in CMS/DOS. Also, COBOL programs that use ISAM or segmentation cannot be executed under CMS/DOS.

WORK FILE REQUIREMENTS

This section describes the work files required by the compiler. In addition to SYSRES and SYSLNK, which must be assigned to disk units, the compiler may require up to six work files: SYS001, SYS002, SYS003, SYS004, SYS005, and SYS006. SYS005 is required only when the Symbolic Debug feature is requested. SYS006 is required for FIPS. Note: SYSLNK is not supported and must not be assigned for CMS/DOS.

SYS001 and SYS006 must be assigned to a disk unit. Each of the other data sets (SYS002, SYS003, SYS004, and SYS005) can be assigned to either a tape unit or a disk unit.

If the work files are on tape, each must be assigned to a separate tape unit. If the work files are on disk, the amount of disk work space required depends mainly on the size of the source program. The approximate percentages of tracks that should be assigned to each work file are shown in Figures 1 through 4. Estimates are for programs of approximately 1000 and 2100 source records. The programs are assumed to request a cross-reference listing, the symbolic debug feature, and the source program library facility (COPY and/or BASIS). Source programs with the library facility require the LIB option on the COBOL option card (CBL card). Programs using NOLIB require considerably less SYS004 work space.

For DOS/VS, work file assignments for the compiler must be made in each partition in which the compiler operates. For example, if the compiler can operate in either the background or the foreground 1 (F1) partition, work file assignments must be made in both the background and F1. The CMS/DOS environment simulates only the background partition. CMS/DOS does not support the 2311.

Number of Source Records	Total Tracks	SYS001 %	SYS002 %	SYS003 %	SYS004 %	SYS005 %
1000	92	22	24	14	34	6
2100	146	16	19	14	44	7

Figure 1. Approximate Allocation of Work File Space -- IBM 2311

Number of Source Records	Total Tracks	SYS001 %	SYS002 %	SYS003 %	SYS004 %	SYS005 %
1000	44	21	18	14	38	9
2100	74	16	16	12	48	8

Figure 2. Approximate Allocation of Work File Space -- IBM 2314

Number of Source Records	Total Tracks	SYS001 %	SYS002 %	SYS003 %	SYS004 %	SYS005 %
1000	28	18	22	14	39	7
2100	46	15	17	13	48	7

Figure 3. Approximate Allocation of Work File Space -- IBM 3330

Number of Source Records	Total Tracks	SYS001 %	SYS002 %	SYS003 %	SYS004 %	SYS005 %
1000	43	19	19	14	39	9
2100	72	15	17	13	48	7

Figure 4. Approximate Allocation of Work File Space -- IBM 3340

FIPS Work Files

The number of tracks required for SYS006 (the FIPS input file) depends on the size of the source program and compiler options specified. As basic output, each record contains a line of the compilation listing and is 121 characters in length. However, additional space is required whenever each of the following options is specified: SYM, LISTX or CLIST, XREF or SXREF. SYS006 also contains any error messages issued by the compiler.

LIOCS MODULES

The following lists contain the names of the preassembled LIOCS modules used by the compiler. These modules are supplied by IBM as part of the relocatable library when the IBM DOS/VS System is distributed. Be sure that the modules listed are included in the system to which the compiler is added.

Certain preassembled LIOCS modules are required when cataloging IBM components to the core image library. Do not delete these modules from the relocatable library until after all the IBM components have been cataloged to the core image library and all program products have been installed. These modules are listed in "Required IOCS Modules."

Notes:

1. Additional LIOCS modules (for ASCII processing) are distributed with the DOS/VS subroutines and are added to the relocatable library with the subroutines. These LIOCS modules are listed under "Relocatable Library" in the "Storage Requirements" section.

2. CMS/DOS does not support all the DOS/VS logical transients, nor all the operands of the transients it does support. The CMS/DOS logical transient support is described in the *VM/370 CMS User's Guide*, Order No. GC20-1819.

The procedure for generating the compiler for use with CMS/DOS is exactly the same as the procedure for generating it for use with DOS/VS: the LIOCS modules required are the same. CMS/DOS, during its open routines, checks that the file being opened is supported by CMS/DOS.

Relocatable Module Naming Conventions

Each module has an 8-character name. The name consists of a 3-character prefix and a 5-character field corresponding to the option permitted in generation of the module. The following 3-character prefixes identify the preassembled modules shipped by IBM:

IJC	I/O Card (CDMOD)
IJD	I/O Printer (PRMOD)
IJF	I/O Magnetic Tape (MTMOD)
IJG	Sequential Direct Access (SDMOD)
IJH	Index Sequential Direct Access (ISMOD)[1]
IJI	Direct Access Method (DAMOD)[1]
IJJ	Device Independent Access Method (DIMOD)
IJN	3540 Access Method (DUMOD)[1]

----

[1] ISMOD, DAMOD, and DUMOD are not supported for the CMS/DOS environment.

CDMOD

CDMOD name = IJCabcde

a	= F	RECFORM=FIXUNB (always for INPUT and CMBND files)
	= U	RECFORM=UNDEF
	= V	RECFORM=VARUNB
b	= A	CTLCHR=ASA (not specified CMBND)
	= C	CCNTROL=YES
	= Y	CTLCHR=YES
	= Z	neither CTLCHR nor CONTROL is specified
c	= B	RDONLY=YES and TYPEFLE=CMBND
	= C	TYPEFLE=CMBND
	= H	RDONLY=YES and TYPEFLE=INPUT
	= I	TYPEFLE=INPUT
	= N	RDONLY=YES and TYPEFLE=OUTPUT
	= O	TYPEFLE=OUTPUT
d	= B	WORKA=YES and IOAREA2=YES
	= I	IOAREA2=YES
	= W	WORKA=YES
	= Z	neither WORKA nor IOAREA2 is specified
	= Z	WORKA is <u>not</u> specified (for CMBND files only)
e	= 0	DEVICE=2540
	= 1	DEVICE=1442
	= 2	DEVICE=2520
	= 3	DEVICE=2501
	= 4	DEVICE=2540 and CRDERR is specified
	= 5	DEVICE=2520 and CRDERR is specified
	= 6	DEVICE=3505
	= 7	DEVICE=3525 and FUNC omitted or FUNC=R or P
	= 8	DEVICE=2560 and FUNC omitted or FUNC=R or P
	= 9	DEVICE=5425 and FUNC omitted or FUNC=R or P
	= A	DEVICE=3525 and FUNC=RP
	= B	DEVICE=3525 and FUNC=RW
	= C	DEVICE=3525 and FUNC=PW
	= D	DEVICE=3525 and FUNC=I
	= E	DEVICE=3525 and FUNC=RPW
	= F	DEVICE=2560 and FUNC=RP
	= G	DEVICE=2560 and FUNC=RW
	= H	DEVICE=2560 and FUNC=PW
	= I	DEVICE=2560 and FUNC=I
	= J	DEVICE=2560 and FUNC=RPW
	= K	DEVICE=5425 and FUNC=RP
	= L	DEVICE=5425 and FUNC=RW
	= M	DEVICE=5425 and FUNC=PW
	= N	DEVICE=5425 and FUNC=I
	= O	DEVICE=5425 and FUNC=RPW

IJCFAOI1
IJCFAOI4
IJCFAOI5
IJCFAOI7
IJCFAOI8
IJCFAOI9
IJCFAOZ1
IJCFAOZ4

IJCFAOZ5
IJCFAOZ7
IJCFAOZ8
IJCFAOZ9
IJCFAOZA
IJCFAOZC
IJCFAOZE
IJCFAOZF
IJCFAOZH
IJCFAOZI
IJCFAOZJ
IJCFAOZK
IJCFAOZM
IJCFAOZN
IJCFAOZO
IJCFCIZ0
IJCFCIZ1
IJCFCIZ2
IJCFCIZ6
IJCFYOI0
IJCFYOI7

IJCFZII0
IJCFZII1
IJCFZII2
IJCFZII3
IJCFZII6
IJCFZII7
IJCFZII8
IJCFZII9
IJCFZIZ0

IJCFZIZ1
IJCFZIZ2
IJCFZIZ3
IJCFZIZ6
IJCFZIZ7
IJCFZIZ8
IJCFZIZ9
IJCFZIZA
IJCFZIZB
IJCFZIZE
IJCFZIZF
IJCFZIZG
IJCFZIZJ
IJCFZIZK
IJCFZIZL
IJCFZIZO
IJCFZOI1
IJCFZOI4

IJCFZOI5
IJCFZOI7
IJCFZOI8
IJCFZOI9
IJCFZOID
IJCFZOZ1
IJCFZOZ4
IJCFZOZ5
IJCFZOZ7
IJFCZOZ8
IJCFZOZ9
IJCFZOZA
IJCFZOZC
IJCFZOZD
IJCFZOZE
IJCFZOZF
IJCFZOZH

IJCFZOZI
IJCFZOZJ
IJCFZOZK
IJCFZOZM
IJCFZOZN
IJCFZOZO

IJCUAOI4
IJCUAOI5
IJCUAOI8
IJCUAOI9

IJCUAOZ1
IJCUAOZ4
IJCUAOZ5
IJCUAOZ8
IJCUAOZ9
IJCUAOZF
IJCUAOZH
IJCUAOZI
IJCUAOZJ
IJCUAOZK
IJCUAOZM
IJCUAOZN
IJCUAOZO

IJCUZII8
IJCUZII9
IJCUZIZ8
IJCUZIZ9
IJCUZIZF
IJCUZIZG
IJCUZIZJ
IJCUZIZK
IJCUZIZL
IJCUZIZO

IJCUZOI1
IJCUZOI4

IJCUZOI5
IJCUZOI7
IJCUZOI8

IJUCZOI9
IJUCZOZ1
IJUCZOZ4
IJUCZOZ5
IJUCZOZ7
IJUCZOZ8
IJUCZOZ9
IJUCZOZF
IJUCZOZH
IJUCZOZI
IJUCZOZJ
IJUCZOZK
IJUCZOZM

IJUCZOZN
IJUCZOZO

IJCVAOI1
IJCVAOI4

IJCVAOI5
IJCVAOI8
IJCVAOI9
IJCVAOZ1
IJCVAOZ4
IJCVAOZ5
IJCVAOZ8
IJCVAOZ9

IJCVAOZF
IJCVAOZH
IJCVAOZI
IJCVAOZJ
IJCVAOZK
IJCVAOZM
IJCVAOZN
IJCVAOZO

IJCVZII8
IJCVZII9
IJCVZIZ8
IJCVZIZ9
IJCVZIZF
IJCVZIZG
IJCVZIZJ
IJCVZIZK
IJCVZIZL
IJCVZIZO

IJCVZOI1
IJCVZOI4
IJCVZOI5
IJCVZOI7
IJCVZOI8

IJCVZOI9

IJCVZOZ1
IJCVZOZ4
IJCVZOZ5
IJCVZOZ7
IJCVZOZ8
IJCVZOZ9
IJCVZOZF
IJCVZOZH
IJCVZOZI
IJCVZOZJ
IJCVZOZK
IJCVZOZM
IJCVZOZN
IJCVZOZO

DAMOD

DAMOD name = IJIabcde

a =	B	RECFORM=UNDEF (handles both UNDEF and FIXUNB)
=	F	RECFORM=FIXUNB
=	S	RECFORM=SPNUNB
=	V	RECFORM=VARUNB
b =	A	AFTER=YES
=	Z	AFTER is <u>not</u> specified
c =	E	IDLOC=YES and FEOVD=YES
=	I	IDLOC=YES
=	R	FEOVD=YES
=	Z	neither is specified
d =	H	ERREXT=YES and RELTRK=YES
=	P	ERREXT=YES
=	R	RELTRK=YES
=	Z	neither is specified
e =	W	HOLD=YES and RDONLY=YES
=	X	HOLD=YES
=	Y	RDONLY=YES
=	Z	neither is specified

DAMOD Names

IJIBAIRZ
IJIBAIZZ
IJIBAZRZ
IJIBAZZZ
IJIBZIRZ

IJIBZIZZ
IJIBZZRZ
IJIBZZZZ
IJIFAIRZ
IJIFAIZZ

IJIFAZRZ
IJIFAZZZ
IJIFZIRZ
IJIFZIZZ
IJIFZZRZ

IJIFZZZZ
IJISAIRZ
IJISAIZZ
IJISAZRZ
IJISAZZZ

IJISZIRZ
IJISZIZZ
IJISZZRZ
IJISZZZZ

DRMOD

DRMOD name = IJMZabD0

a = S SETDEV = YES
  = Z SETDEV = NO

b = R RDONLY = YES
  = Z RDONLY = NO

DRMOD Names
IJMZSRD0
IJMZSZD0
IJMZZRD0
IJMZZZD0

DUMOD

DUMOD name = IJNDabcZ

a = I input
  = O output

b = C ERROPT=name

c = Z RDONLY not specified

DUMOD Names
IJNDICZZ
IJNDOCZZ

ISMOD

ISMOD name = IJHabcde

```
a = A  RECFORM=BOTH and IOROUT=ADD or ADDRTR
  = B  RECFORM=FIXBLK and IOROUT=ADD or ADDRTR
  = U  RECFORM=FIXUNB and IOROUT=ADD or ADDRTR
  = Z  RECFORM is not specified and IOROUT=LOAD or RETRVE

b = A  IOROUT=ADDRTR
  = I  IOROUT=ADD
  = L  ICROUT=LOAD
  = R  IOROUT=RETRVE

c = B  TYPEFLE=RANSEQ
  = G  IOAREA2=YES and TYPEFLE=SEQNTL or IOROUT=LOAD
  = R  TYPEFLE=RANDOM
  = S  TYPEFLE=SEQNTL
  = Z  neither is specified and IOROUT=LOAD or ADD

d = C  CORINDX=YES
  = Z  CORINDX is not specified

e = F  CORDATA=YES, ERREXT=YES, and RDONLY=YES
  = G  CORDATA=YES and ERREXT=YES
  = O  CORDATA=YES and RDONLY=YES
  = P  CORDATA=YES
  = S  ERPEXT=YES and RDONLY=YES
  = T  ERREXT=YES
  = Y  RDONLY=YES
  = Z  nothing is specified
```

ISMOD Names

IJHAARCP
IJHAARCZ
IJHAARZP
IJHAARZZ
IJHBABCP

IJHBABCZ
IJHBABZP
IJHBABZZ
IJHBARCP
IJHBARCZ

IJHBARZP
IJHBARZZ
IJHUABCP
IJHUABCZ
IJHUABZP

IJHUABZZ
IJHUARCP
IJHUARCZ
IJHUARZP
IJHUARZZ

IJHZLGZZ
IJHZLZZZ
IJHZRBCZ
IJHZRBZZ

IJHZRGZZ
IJHZRRZZ
IJHZRSZZ

MTMOD

MTMOD name = IJFabcde

```
a = F  RECFORM=FIXUNB or FIXBLK
  = S  RECFORM=SPNUNB or SPNBLK
  = U  RECFORM=UNDEF
  = V  RECFORM=VARUNB or VARBLK

  = X  ASCII RECFORM=FIXUNB or FIXBLK
  = N  ASCII RECFORM=UNDEF
  = R  ASCII RECFORM=VARUNB or VARBLK

b = B  READ=BACK
  = Z  READ=FORWARD, or READ is not specified

c = C  CKPTREC=YES
  = Z  CKPTREC is not specified

d = W  WORKA=YES
  = Z  WORKA is not specified

e = M  ERREXT=YES and RDONLY=YES
  = N  ERREXT=YES
  = Y  RDONLY=YES
  = Z  neither is specified
```

MTMOD Names (See "Relocatable Library" in the "Storage Estimates" section for additional MTMOD modules.)

IJFFBZZN
IJFFZCZZ
IJFFZZZZ
IJFSZZWN
IJFUZZZN

IJFUZZZZ
IJFVZZZN
IJFVZZZZ

Name list for workfile type modules (TYPEFLE=WORK):

MTMOD name = IJFabcde

```
a = W  always

b = E  ERROPT=YES
  = Z  ERROPT is not specified

c = N  NOTEPNT=YES
  = S  NCTEPNT=POINTS
  = Z  NOTEPNT is not specified

d = Z  always

e = M  ERREXT=YES and RDONLY=YES
  = N  ERREXT=YES
  = Y  RDONLY=YES
  = Z  neither is specified
```

System I/O Modules
(See "Required IOCS Modules" before deleting modules with the IJF prefix.)

IJFWEZZZ
IJFWZNZZ
IJFWZZZZ

PRMOD

PRMOD name = IJDabcde

```
a = F  RECFORM=FIXUNB
  = V  RECFORM=VARUNB
  = U  RECFORM=UNDEF

b = A  CTLCHR=ASA
  = Y  CTLCHR=YES
  = C  CONTROL=YES
  = S  STLIST=YES
  = T  DEVICE=3525 with 2-line printer
  = U  DEVICE=2560
  = V  DEVICE=5425
  = Z  neither CTLCHR nor CONTROL nor STLIST is specified

c = P  PRINTOV=YES, DEVICE not 3525
  = I  PRINTOV=YES, DEVICE=3525, and FUNC Omitted or FUNC=W[T]
  = F  PRINTOV=YES, DEVICE=3525, and FUNC=RW[T]
  = C  PRINTOV=YES, DEVICE=3525, and FUNC=PW[T]
  = D  PRINTOV=YES, DEVICE=3525, and FUNC=RPW[T]
  = E  ERROPT=YES and PRINTOV=YES not specified
  = Z  PRINTOV=YES not specified, and DEVICE not 3525
  = O  PRINTOV=YES not specified, DEVICE=3525, and FUNC omitted
       or FUNC=W[T]
  = R  PRINTOV=YES not specified, DEVICE=3525, and FUNC=RW[T]
  = S  PRINTOV=YES not specified, DEVICE=3525, and FUNC=PW[T]
  = T  PRINTOV=YES not specified, DEVICE=3525, and FUNC=RPW[T]
  = U  DEVICE=2560 or 5425 and FUNC=W or omitted
  = V  DEVICE=2560 or 5425 and FUNC=RW
  = W  DEVICE=2560 or 5425 and FUNC=PW
  = X  DEVICE=2560 or 5425 and FUNC=RPW

d = I  ICAREA2=YES
  = Z  IOAREA2 is not specified

e = V  RDONLY=YES and WORKA=YES
  = W  WORKA=YES
  = Y  RDONLY=YES
  = Z  neither is specified
```

PRMOD Names

IJDFAPIZ
IJDFAPZZ
IJDFYPIZ
IJDFYPZZ
IJDFZPIZ

IJDFZPZZ
IJDUAPIZ
IJDUAPZZ
IJDUYPIZ
IJDUYPZZ

IJDUZPIZ
IJDUZPZZ
IJDVAPIZ
IJDVAPZZ
IJDVYPIZ

IJDVYPZZ
IJDVZPIZ
IJDVZPZZ

IJDFUUZZ
IJDFUUIZ
IJDFUVZZ
IJDFUVIZ
IJDFUWZZ
IJDFUWIZ

IJDVUUZZ
IJDVUUIZ
IJDVUVZZ
IJDVUVIZ
IJDVUWZZ
IJDVUWIZ

IJDUUUZZ
IJDUUUIZ
IJDUUVZZ
IJDUUVIZ
IJDUUWZZ
IJDUUWIZ

IJDFUXZZ	IJDVUXZZ	IJDUUXZZ
IJDFUXIZ	IJDVUXIZ	IJDUUXIZ

IJDFAOIZ	IJDVAOIZ	IJDUAOIZ
IJDFARIZ	IJDVARIZ	IJDUARIZ
IJDFASIZ	IJDVASIZ	IJDUASIZ
IJDFATIZ	IJDVATIZ	IJDUATIZ

IJDFAOZZ	IJDVAOZZ	IJDUAOZZ
IJDFARZZ	IJDVARZZ	IJDUARZZ
IJDFASZZ	IJDVASZZ	IJDUASZZ
IJDFATZZ	IJDVATZZ	IJDUATZZ
IJDFZOIZ	IJDVZOIZ	IJDUZOIZ
IJDFZRIZ	IJDVZRIZ	IJDUZRIZ
IJDFZSIZ	IJDVZSIZ	IJDUZSIZ
IJDFZTIZ	IJDVZTIZ	IJDUZTIZ
IJDFZOZZ	IJDVZOZZ	IJDUZOZZ
IJDFZRZZ	IJDVZRZZ	IJDUZRZZ
IJDFZSZZ	IJDVZSZZ	IJDUZSZZ
IJDFZTZZ	IJDVZTZZ	IJDUZTZZ

IJDFAIIZ	IJDUAIIZ	IJDVAIIZ
IJDFAFIZ	IJDUAFIZ	IJDVAFIZ
IJDFACIZ	IJDUACIZ	IJDVACIZ
IJDFADIZ	IJDUADIZ	IJDVADIZ

IJDFZIIZ	IJDUZIIZ	IJDVZIIZ
IJDFZFIZ	IJDUZFIZ	IJDVZFIZ
IJDFZCIZ	IJDUZCIZ	IJDVZCIZ
IJDFZDIZ	IJDUZDIZ	IJDVZDIZ

IJDFYIIZ	IJDUYIIZ	IJDVYIIZ
IJDFYFIZ	IJDUYFIZ	IJDVYFIZ
IJDFYCIZ	IJDUYCIZ	IJDVYCIZ
IJDFYDIZ	IJDUYDIZ	IJDVYDIZ

IJDFVUZZ	IJDVVUZZ	IJDUVUZZ
IJDFVUIZ	IJDVVUIZ	IJDUVUIZ
IJDFVVZZ	IJDVVVZZ	IJDUVVZZ
IJDFVVIZ	IJDVVVIZ	IJDUVVIZ
IJDFVWZZ	IJDVVWZZ	IJDUVWZZ
IJDFVWIZ	IJDVVWIZ	IJDUVWIZ
IJDFVXZZ	IJDVVXZZ	IJDUVXZZ
IJDFVXIZ	IJDVVXIZ	IJDUVXIZ

SDMOD

SDMODxx name = IJGabcde

```
a = C  RECFORM=FIXUNB or FIXBLK and HOLD=YES
  = F  RECFORM=FIXUNB or FIXBLK and HOLD is not specified
  = P  RECFORM=SPNUNB or SPNBLK and HOLD=YES
  = Q  RECFORM=SPNUNB or SPNBLK and HOLD is not specified
  = R  RECFORM=UNDEF and HOLD=YES
  = S  RECFORM=VARUNB or VARBLK and HOLD=YES
  = U  RECFORM=UNDEF and HOLD is not specified
  = V  RECFORM=VARUNB or VARBLK and HOLD is not specified

b = I  SDMODxI
  = O  SDMODxO
  = U  SDMODxU

c = C  ERROPT=YES and ERREXT=YES
  = E  ERROPT=YES
  = Z  neither is specified

d = M  TRUNCS=YES and FEOVD=YES
  = T  TRUNCS=YES
  = W  FEOVD=YES
  = Z  neither is specified

e = B  CONTROL=YES and RDONLY=YES
  = C  CONTROL=YES
  = Y  RDONLY=YES
  = Z  neither is specified
```

SDMOD Names
IJGFIEWZ
IJGFOEWZ
IJGFUEWZ
IJGQIEWZ
IJGQOEWZ
IJGQUEWZ
IJGUIEWZ
IJGUOEWZ
IJGUUEWZ
IJGVIEWZ
IJGVOEWZ
IJGVUEWZ

NAME LIST FOR WORKFILE TYPE MODULES (TYPEFLE=WORK)

```
SDMODxx name = IJGabcde

a = T  SDMODW specifies HOLD=YES
  = W  SDMODW does not specify HOLD=YES

b = C  ERROPT=YES and ERREXT=YES
  = E  ERROPT=YES
  = Z  neither is specified

c = N  NOTEPNT=YES
  = R  NOTEPNT=PCINTRW
  = Z  NOTEPNT is not specified

d = C  CONTROL=YES
  = Z  CONTROL is not specified

e = T  RDONLY=YES and UPDATE=YES
  = U  UPDATE=YES
  = Y  RDONLY=YES
  = Z  neither is specified
```

System I/O Modules

(See "Required IOCS Modules" before deleting modules with the IJG prefix.)

IJGWEZZU
IJGWEZZZ
IJGWZNZZ
IJGWZRZZ

Required IOCS Modules

The following preassembled IOCS modules are required when cataloging IBM components to the core image library. These modules are also required when installing program products.

Module Names

IJFWEZZZ
IJFWZNZZ
IJFWZZZZ
IJGFIETZ
IJGWEZZU

IJGWEZZZ
IJGWZNZZ
IJGWZRZZ
IJJCPA1N
IJJCPDV

IJJCPDV1
IJJCPDV2
IJJCPD0
IJJCPD0N
IJJCPD1

IJJCPD1N
IJJCPD2
IJJCPD3
IJJCPV
IJJCPV1

IJJCPV2
IJJCP0
IJJCP0N
IJJCP1
IJJCP1N

IJJCP2
IJJCP3

INSTALLATION PROCEDURE

To add the DOS/VS Compiler and the Library to your system, follow the procedure described in this section. The method used to install the compiler and the library depends on the medium on which each was received. For the procedures pertinent to your installation, refer to the appropriately named section below.

- Installation from Tape

- Separate Library Installation from Tape

- Installation from Disk

- Separate Library Installation from Disk

The system to which the compiler and library are to be added must include the required IOCS modules (listed in "IOCS Modules") as well as the DSTRB and Maintenance utility programs. System generation factors that can affect compiler execution are discussed below. If the compiler and library are to be used in a foreground partition, special preparation is necessary. See "Foreground Installation Considerations" for the steps required.

Note: The CMS/DOS environment simulates only the background partition. Therefore, the compiler must be generated for use in the background partition with ACTION REL if it is going to be used in CMS/DOS.

Preparing the DOS/VS System Libraries

Before actually installing the compiler and/or library subroutines, you may have to prepare the DOS/VS system libraries. During installation, the compiler and subroutines replace any IBM American National Standard COBOL Compiler and Subroutine Library already existing in the library to which they are added. Therefore, to preserve an existing version of the IBM American National Standard COBOL Compiler and Subroutine Library, you must move them to a private library before adding the DOS/VS compiler and subroutines to the system libraries or add the DOS/VS compiler and subroutines to a private library during installation.

To replace an existing version, use the "Storage Requirements" section to determine if sufficient space is available for the DOS/VS compiler and subroutines. If there is enough space in the library, no preparation for installation is necessary. If not, the existing version should be deleted and the libraries condensed. The following examples show how to delete an earlier compiler and subroutine library and then condense the system libraries. These jobs must be run in the background.

```
// JOB    DELETE
// EXEC   MAINT
   DELETC FCOB.ALL
   DELETC ILBD.ALL
   DELETC $$BCOBER
   DELETC $$BCOBR1
   DELETC $$BFCMUL
   DELETC $$BCOBEM
   DELETR ILA.ALL
   DELETR ILB.ALL
/*
/&

// JOB    CONDENSE
// EXEC   MAINT
   CONDS  CL
   CONDS  RL
/*
/&
```

The output from this job will indicate the number of available blocks remaining in each library.

System Generation Considerations

The compiler and library can be used on a system only if the following parameters are specified for the SUPVR and FOPT macro instructions during system generation:

SUPVR:	SYSTEM=DISK	
	ASCII=YES	(required only when the ASCII features of the compiler are used)
	MPS=BJF	(required only when the compiler is to be executed in the foreground)
FOPT:	AB=YES	(for use of STATE, FLOW, STXIT, SYMDMP, and COUNT features)
	PCIL=YES	(required only when the compiler is to be executed in the foreground)
	GETVIS=YES	(required for use of 3886; forced by VSAM=YES, and required if COUNT option is specified)
VSAM:	VSAM=YES	(required for use of the VSAM feature)

If PCIL=YES is specified with MPS=NO or MPS=YES, private core image library support is provided for background only. When MPS=BJF is specified, a private core image library can be assigned either to the background or to a foreground partition; it cannot be assigned to more than one partition.

FOPT=SYSFIL must also have been specified during system generation if system files on disk are to be used.

System options selected at system generation enable you to tailor the DOS/VS compiler to fit your installation's needs. The system options that can be used to control compiler processing specify whether:

- Control statements are to be written on SYSLST.
- A dump is to be written on SYSLST if an abnormal termination occurs. (You may not want this if the SYMDMP, STATE, or FLOW features are used.)

- The object module produced by the compiler is to be link edited.

- An object deck is to be punched.

- The COBOL source statements are to be written on SYSLST.

- A Procedure Division map, Data Division map, or cross-reference listing is to be written on SYSLST.

- Diagnostic messages for the source program are to be written on SYSLST.

Instructions for specifying these options are given in the publications DOS/VS System Generation and DOS/VS COBOL Compiler and Library Programmer's Guide.

CHANGING THE INSTALLATION DEFAULTS

To change the compiler default values to suit your installation, a new member, C.CBLOPTNS, must be added to the source statement library. This module must contain CBL and LST option cards defining the desired defaults. These may be overridden at compilation time by supplying a CBL and/or LST statement in the compiler input stream. CMS/DOS does not support the LST statement.

CBL Statement -- COBOL Option Control Statement

Although most options for compilation are specified either at system generation time or in the OPTION control statement, the COBOL compiler provides an additional statement, the CBL statement, for the specification of compile-time options unique to COBOL.

The CBL statement must be placed between the EXEC FCOBOL statement and the first statement in the COBOL program. The CBL statement cannot be continued. However, if specification of options will continue past column 71, more than one CBL card image may be used.

The options shown in the following format may appear in any order. No embedded blanks may appear in the operand field. Underscoring indicates the default value. No comments should appear in the operand field.

```
                    ┌,SEQ   ┐ ┌,FLAGW┐ ┌,SUPMAP   ┐              ┌,LVL=c ┐
CBL [BUF=nnnn]      └,NOSEQ ┘ └,FLAGE┘ └,NOSUPMAP ┘  [,SPACEn]   └,NOLVL ┘

 ┌,CLIST   ┐   ┌,STXIT   ┐┌,QUOTE┐ ┌,TRUNC   ┐ ┌,ZWB  ┐   ┌,SYNTAX   ┐
 └,NOCLIST ┘   └,NOSTXIT ┘└,APOST┘ └,NOTRUNC ┘ └,NOZWB┘   │,CSYNTAX  │
                                                          └,NOSYNTAX ┘

 ┌,SXREF   ┐              ┌,OPTIMIZE   ┐                    ┌,STATE   ┐
 └,NOSXREF ┘   [,PMAP=h]  │,NOOPTIMIZE │     [,FLOW[=nn]]   └,NOSTATE ┘
                          │,OPT        │
                          └,NOOPT      ┘

  [,SYMDMP[=filename]]    ┌,CATALR   ┐      ┌,LIB   ┐        ┌,VERB   ┐
                          └,NOCATALR ┘      └,NOLIB ┘        └,NOVERB ┘

 ┌,VERBSUM   ┐  ┌,VERBREF   ┐┌,COUNT   ┐
 └,NOVERBSUM ┘  └,NOVERBREF ┘└,NOCOUNT ┘
```

CBL

must begin in column 2 (column 1 must be blank) and be followed by at least one blank.

BUF=nnnnn

the BUF option specifies the amount of storage to be assigned to each compiler work file buffer (i.e., the block sizes of the work files). nnnnn is a decimal number from 512 to 32,767. If this option is not specified, 512 is assumed. Performance is affected by the use of this option, and the amount specified should be tailored for best performance for the work file devices used. Maximum performance is usually achieved by specifying the buffers as large as possible.

SEQ
NOSEQ

indicates whether or not the compiler is to check the sequence of source statements. If SEQ is specified and a statement is not in sequence, it is flagged. If lister is invoked also, the original card sequence numbers may be resequenced, thus not providing the sequence check desired.

FLAGW
FLAGE

determines which diagnostic messages the compiler will list. FLAGW indicates that all diagnostic messages will be listed (severity levels W, C, E, and D). FLAGE indicates that only those diagnostic messages with severity levels C, E, and D will be listed. This has no effect on FIPS messages.

SUPMAP
NOSUPMAP

causes the CLIST and LISTX options to be suppressed if an E-level diagnostic message is produced by the compiler. SUPMAP also suppresses the DECK option, and no object module is produced.

SPACEn

indicates the line of spacing to be used on the output listing.

n

can be specified as 1 (single spacing), 2 (double spacing), or 3 (triple spacing). If the SPACEn option is omitted, single spacing is provided.

LVL=c
LVL=NO

indicates whether the compiler should identify COBOL clauses and statements in a DOS/VS COBOL source program that do not conform to the Federal Information Processing Standard. FIPS recognizes four language levels: low, low-intermediate, high-intermediate and full. The FIPS Flagger provides four levels of flagging frcm low (A) to high (D) to conform to the four levels of the FIPS.

CLIST
NOCLIST

indicates that a condensed listing is to be produced. The condensed listing contains only the address of the first generated instruction for each verb in the Procedure Division. In addition, global tables, literal pools, register assignments, and procedure block assignments are provided. The CLIST option overrides the LISTX or NOLISTX options. The LISTX or NOLISTX options are either established at system generation time or specified in the OPTION control statement.

STXIT
NOSTXIT

enables a USE AFTER STANDARD ERROR declarative to receive control when an input/output error occurs on a unit record device. The use of STXIT precludes the use of SYMDMP, STATE, and FLOW in the compiled program and in any other program link-edited with the compiled program, and vice versa.

QUOTE
APOST

QUOTE indicates to the compiler that the double quotation marks (") should be accepted as the character to delineate literals; APOST indicates that the apostrophe (') should be accepted instead. The compiler will generate the specified character for the figurative constant QUOTE(S).

TRUNC
NOTRUNC

applies only to COMPUTATIONAL receiving fields in MOVE statements and arithmetic expressions. If TRUNC is specified, extra code is generated to truncate the final intermediate result of the arithmetic expression, or the sending field in the MOVE statement, to the number of digits specified in the PICTURE clause of the COMPUTATIONAL receiving field. If NOTRUNC is specified, the compiler assumes that the data being manipulated conforms to PICTURE and USAGE specifications. The compiler then generates code to manipulate the data based on the size of the field in core (halfword, etc.). TRUNC conforms to the American National Standard, while NOTRUNC leads to more efficient processing. This will occasionally cause dissimilar results for various sending fields because of the different code generated to perform the operation.

ZWB

NOZWB

determines if the compiler is to generate code to strip the sign when comparing a signed external decimal field to an alphanumeric field. If ZWB is in effect, the signed external decimal field is moved to an intermediate field and has its sign stripped before being compared to the alphanumeric field. ZWB conforms to the American National Standard, while NOZWB allows the user to test input numeric fields for SPACES to prevent abnormal termination.

SYNTAX

CSYNTAX

NOSYNTAX

indicates whether the source text is to be scanned for syntax errors only and appropriate error messages are to be generated. For conditional syntax checking (CSYNTAX), a full compilation is produced so long as no messages exceed the C level. If one or more E-level or higher severity messages are produced, the compiler generates the messages but does not generate object text.

Notes:

1. When the SYNTAX option is in effect, all of the following compile-time options are suppressed:

   OPTION control statement: LINK, DECK, XREF, LISTX

   CBL statement: SXREF, CLIST, COUNT, VERBSUM, VERBREF

2. When conditional syntax-checking is requested, the preceding options are suppressed only if one or more E- or D-level messages are generated.

3. Unconditional syntax checking is assumed if all of the following compile-time options are specified:

   OPTION control statement: NOLINK, NOXREF, NODECK, LISTX

   CBL statement: SUPMAP (and CLIST, SXREF, VERBSUM, and VERBREF are not specified)

4. Some compiler diagnostics do not appear when SYNTAX or CSYNTAX is in effect.

SXREF

NOSXREF

causes the compiler to write an alphabetically-ordered cross-reference list on SYSLST. You may want to use the lister feature cross-reference information instead of this option for large COBOL programs, to decrease run time and get a better type of cross-reference.

PMAP=h

enables the programmer to request a relocation factor "h" specified as one to eight hexadecimal digits. If the PMAP option is specified, the relocation factor is included in the addresses of the object code listing. If the PMAP option is not specified, the relocation factor is assumed to be zero. When PMAP is specified in a segmented program, the listing for segments of priority higher than the segment limit (49, if the SEGMENT-LIMIT clause is not specified) will not be relocated. The PMAP option is effective only if LISTX is specified.

OPTIMIZE
NOOPTIMIZE
OPT
NOOPT

OPTIMIZE (OPT) causes optimized object code to be generated by the compiler. This more efficient code considerably reduces the amount of space required by the object program.

This option cannot be used if either the symbolic debug option (SYMDMP), the statement number option (STATE), or the flow trace option (FLOW[=nn]) is requested.

Note: If NODECK and NOLINK are requested on the OPTION control statement and OPT is specified on the CBL card, OPT is ignored.

FLOW[=nn]

provides the programmer with a formatted trace (i.e., a list containing the program identification and statement numbers) corresponding to a variable number of procedures executed prior to an abnormal termination. The value "nn" may be from 0 to 99. If "nn" is not specified, a value of 99 is assumed.

FLOW and STXIT, and FLOW and OPT are mutually exclusive, i.e., only one may be in effect during a compilation. In addition, FLOW and STXIT are mutually exclusive at execution time. Additional information on the flow trace option can be found in "Symbolic Debug Considerations."

STATE
NOSTATE

STATE provides the programmer with information about the statement being executed at the time of an abnormal termination of a job. It identifies the program containing the statement and provides the number of the statement and of the verb being executed. STATE and STXIT, STATE and SYMDMP, and STATE and OPT are mutually exclusive options, i.e., no more than one may be in effect during a given compilation. (However, the facilities provided by STATE automatically exist with SYMDMP.) In addition, STATE and STXIT are mutually exclusive at executiom time. Additional information on the statement number option can be found in the chapter "Symbolic Debug Considerations."

SYMDMP[=filename]

indicates to the compiler that execution-time dumps may be requested for the program currently being compiled. If dumps are desired, the programmer must provide the required control cards at execution time.

Use of the symbolic debug option necessitates the presence of an additional work file, SYS005, at compile time. The "filename" parameter enables the programmer to specify a name for the SYS005 file that he can retain. If no filename is specified, IJSYS05 will be used. When several COBOL programs are link edited together, the "filename" parameter enables each to have a unique SYMDMP name. For a tape file, only unlabeled tapes may be used, and the filename in the SYMDMP=filename parameter is ignored.

SYMDMP and STXIT, SYMDMP and STATE, and SYMDMP and OPT are mutually exclusive, i.e., only one may be in effect during a compilation. (However, the facilities provided by STATE automatically exist with SYMDMP.) In addition, SYMDMP and STIXIT are mutually exclusive at execution time. Additional information on the symbolic debug option and the required execution-time control cards is in "Symbolic Debug Considerations."

Note: If NODECK and NOLINK are requested on the OPTION control statement and SYMDMP is specified on the CBL card, SYMDMP is ignored.

CATALR
NOCATALR

causes the compiler to generate CATALR card images on the SYSPCH file if OPTION DECK is in effect during compilation. This will allow cataloging of the compiler-produced object modules into the relccatable library. The module names in the CATALR cards follow the same rules as the phase names in the compiler-produced PHASE cards according to the segmentation and sort phase naming conventions. (See "Sort Feature Considerations.")

LIB
NOLIB

indicates that BASIS and/or COPY statements are in the source program. If either COPY or BASIS is present, LIB must be in effect. If COPY and/or BASIS statements are not present, use of the NOLIB option yields more efficient compiler processing.

VERB
NOVERB

indicates whether procedure-names and verb-names are to be listed with the associated code on the object program listing. VERB has meaning only if the LISTX, VERBSUM, VERBREF, COUNT, or CLIST compiler option is specified or if READY TRACE is used in the source program. NOVERB yields more efficient compilation.

VERBSUM
NOVERBSUM

provides a brief summary of verbs used in the program and a count of how often each verb is used. This option provides the user with a quick search for specific types of statements. VERBSUM implies VERB.

VERBREF
NOVERBREF

provides a cross-reference of all verbs used in the program. This option provides the programmer with a quick index to any verb used in the program. VERBREF implies VERB and VERBSUM.

COUNT
NOCOUNT

generates code to produce verb execution summaries at the end of problem program execution. Each verb is identified by procedure-name and by statement number, and the number of times it was used is indicated. In addition, the percentage of verb execution for each verb with respect to the execution of all verbs is given. A summary of all verbs used in a program and the number of times they are executed is provided. COUNT implies VERB. COUNT requires GETVIS=YES, which means that the SIZE parameter must be specified on the EXEC card.

Note: If COUNT and STXIT are desired, then either STXIT must be requested in the program unit requesting COUNT, or the program unit requesting COUNT must be entered before the program unit requesting STXIT.

MUTUALLY EXCLUSIVE OPTIONS: In some of the preceding descriptions of the CBL card options, restrictions have been placed on the use of one option in conjunction with others. If these restrictions are violated, the compiler ignores all but the last of the conflicting options specified. Therefore, if after a CBL card is coded you decide to use a new option that is mutually exclusive with an option on the original CBL card, a new CBL card can be added rather than changing the original card.

CHANGING THE INSTALLATION DEFAULTS: In order to change the compiler default options to suit your installation, a new member, C.CBLOPTNS, must be added to the source statement library. This module must contain CBL option cards specifying the desired defaults. Resultant defaults may be overridden at compilation time by supplying a CBL card in the compiler input stream.

LST Statement -- Compiler Option Statement

The LST statement is used to invoke the lister, a portion of the compiler that processes programs written in DOS/VS COBOL, to produce a reformatted source code listing containing embedded cross-reference information and uniform indenting conventions.

The LST option statement can be placed anywhere between the EXEC statement and the first statement of the COBOL program. It may be placed between any other compiler option statements. The options shown in the following format may appear in any order. Underscoring indicates the default case.

$$\text{LST} \begin{bmatrix}\text{DECK,}\\ \underline{\text{NODECK}}\end{bmatrix} \quad \begin{bmatrix}\text{COPYPCH,}\\ \underline{\text{NOCOPYPCH}}\end{bmatrix} \quad , \begin{bmatrix}\text{LSTCOMP,}\\ \underline{\text{LSTONLY}}\end{bmatrix}$$

$$\begin{bmatrix}\text{PROC=}\underline{\text{1col}}\text{,}\\ \text{2col}\end{bmatrix}$$

LST
must begin in column 2 (column 1 must be blank) and be followed by at least one blank.

DECK
NODECK
indicates whether an updated source deck is to be produced as a result of the lister reformatting and/or the update BASIS library.

COPYPCH
NOCOPYPCH
will punch updated and reformatted copy libraries as a permanent part of the source when DECK is specified. When no updated source deck is requested, an updated and reformatted COPY library will be punched out.

LSTONLY
LSTCOMP
When LSTONLY is specified, the program will not be compiled, but a reformatted listing will be produced along with a deck if DECK has

been specified. LSTCOMP will provide a source listing and will compile the program as part of the job step.

PROC=1col
2col

will list the Procedure Division in either single- or double-column format. At least 132 print positions are required on the printer for the double-column format.

SIGNIFICANT CHARACTERS FOR CBL/LST OPTIONS: The DOS/VS COBOL compiler selects the valid options for processing by looking for three significant characters of each key option word. When the keyword is identified, it is checked for the presence or absence of the prefix NO, as appropriate. The programmer can make the most efficient use of the CBL or LST statement by using the significant characters instead of the entire option. Figure 5 lists the significant characters for each option.

Option	Significant Characters
SEQ	SEQ
FLAGE(W)	LAG,LAGW
BUF	BUF
SPACE	ACE
PMAP	PMA
SUPMAP	SUP
CLIST	CLI
TRUNC	TRU
APOST	APO
QUOTE	QUO
SXREF	SXR
STATE	STA
FLOW	FLO
LIB	LIB
SYMDMP	SYM
OPTIMIZE	OPT
SYNTAX	SYN
CSYNTAX	CSY
VERB	VER
ZWB	ZWB
LVL	LVL
STXIT	STX
VERBSUM	VERBSUM
VERBREF	VERBREF
COUNT	COU
DECK	DEC
COPYPCH	COP
LSTCOMP	STC
LSTONLY	STO
PROC	PRO

Figure 5. Significant Characters for Each CBL/LST Option

Note: SYM in the CBL statement should not be confused with SYM in the OPTION statement.

FOREGROUND INSTALLATION CONSIDERATIONS

In order for the compiler to be link edited in a foreground partition, you may have to create the required private core image library and any additional private libraries required for the partition in which the compiler is to execute. When the private libraries have been created, use the appropriate procedure -- either installation from disk or installation from tape -- to install the compiler and library, specifying the names of the private libraries where requested. The job must be run in the background. If your system has the optional relocating loader, however, you need only a copy of the program in the system core image library.

The following job creates partition standard labels for the background and defines the extents for four system work files to be used in the background. This job must be run in the background.

```
// JOB      LABELS FOR BG
// OPTION   PARSTD
// DLBL     IJSYSRS,'DOS SYSRES FILE',99/365,SD
// EXTENT   SYSRES,111111,1,1,001,489
// DLBL     IJSYSCL,'DOS SYSCLB LIB',99/365,SD
// EXTENT   SYSCLB,111111,1,1,490,700
// DLBL     IJSYSRL,'DOS SYSRLB LIB',99/365,SD
// EXTENT   SYSRLB,111111,1,1,1190,510
// DLBL     IJSYSSL,'DOS SYSSLB LIB',99/365,SD
// EXTENT   SYSSLB,111111,1,1,1700,120
// DLBL     IJSYSRC,'DOS RECORD FILE',99/365,SD
// EXTENT   SYSREC,111111,1,1,1970,10
// DLBL     ESTVFLE,'TAPE ERROR STAT',99/365,SD
// EXTENT   SYS004,111111,1,1,1980,10
// DLBL     IJSYS01,'SYS001 WORK FILE',99/365,SD
// EXTENT   SYS001,222222,8,1,0010,250,4
// DLBL     IJSYS02,'SYS002 WORK FILE',99/365,SD
// EXTENT   SYS002,222222,8,1,0015,250,9
// DLBL     IJSYS03,'SYS003 WORK FILE',99/365,SD
// EXTENT   SYS003,333333,8,1,0010,250,4
// DLBL     IJSYS04,'SYS004 WORK FILE',99/365,SD
// EXTENT   SYS004,333333,8,1,0015,250,9
// DLBL     IJSYSIN,'SYSLNK WORK FILE',99/365,SD
// EXTENT   SYSLNK,111111,1,1,1820,150
/*
/&
```

As a result of the above job, partition standard labels for the background are created, and the extents for the work files (SYS001, SYS002, SYS003, SYS004) highlighted in Figure 6 are defined. The libraries on volser=111111 were created at system generation time.

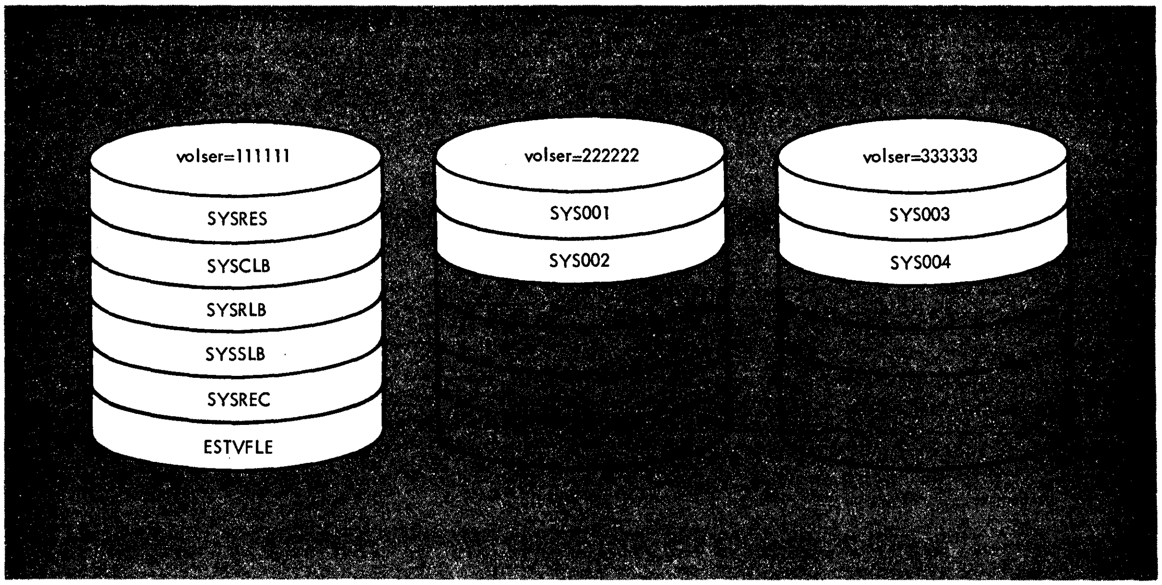

Figure 6. Allocation for Work Files to be Used in Background

To allow for compilation in the foreground, a private core image library assigned in the foreground must be created. In this example, a private relocatable library and a private source statement library are also created, though the system libraries can be used if they are assigned in the foreground.

To create the three private libraries, the following three jobs are run. The first job creates the private core image library; the second creates the private relocatable library; the third creates the private source statement library. These jobs must be run in the background.

```
// JOB      CREATE PCIL
// ASSGN    SYS003,X'cuu'
// DLBL     IJSYSPC,'PRIVATE CORE IMAGE LIB',99/365,SD
// EXTENT   SYS003,333333,1,1,1010,800
// EXEC     CORGZ
   NEWVOL   CL=80(20)
/*
/&

// JOB      CREATE PRL
// ASSGN    SYSRLB,X'cuu'
// DLBL     IJSYSRL,'PRIVATE RELOCATABLE LIB',99/365,SD
// EXTENT   SYSRLB,222222,1,1,1010,750
// EXEC     CORGZ
   NEWVOL   RL=75(25)
/*
/&

// JOB      CREATE PRIVATE SYSSLB
// ASSGN    SYSSLB,X'cuu'
// DLBL     IJSYSSL,'PRIVATE SOURCE LIB',99/365,SD
// EXTENT   SYSSLB,222222,1,1,1760,230
// EXEC     CORGZ
   NEWVOL   SL=23(6)
/*
/&
```

where: cuu is the unit address of the drive on which the disk is mounted.

As a result of these three jobs, the three private libraries are added to the system, as highlighted in Figure 7.

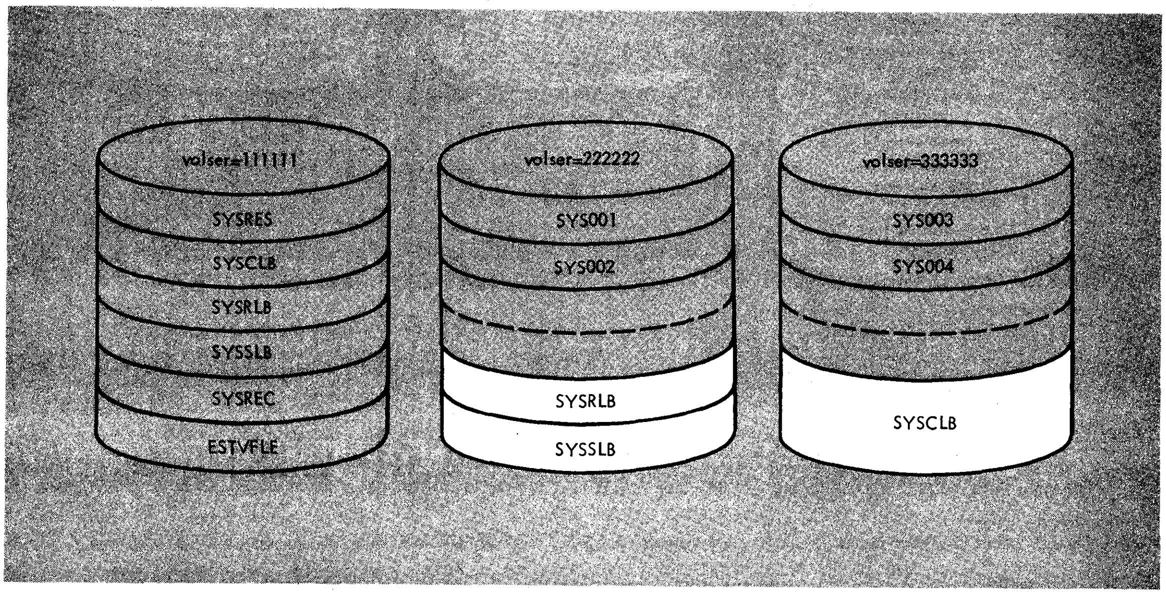

Figure 7. Allocation after Creation of Private Libraries

Next, the following job is run to create partition standard labels for foreground partition 2 (F2) and to define the extents of four system work files to be used in the foreground. This job must be run in the foreground.

```
// JOB      LABELS FOR F2
// OPTION   PARSTD
// DLBL     IJSYSRS,'DOS SYSRES FILE',99/365,SD
// EXTENT   SYSRES,111111,1,1,001,489
// DLBL     IJSYSCL,'PRIVATE CORE IMAGE LIB',99/365,SD
// EXTENT   SYSCLB,333333,1,1,1010,800
// DLBL     IJSYSLN,'SYSLNK WORK FILE',99/365,SD
// EXTENT   SYSLNK,333333,1,1,1810,160
// DLBL     IJSYSRL,'PRIVATE RELOCATABLE LIB',99/365,SD
// EXTENT   SYSRLB,222222,1,1,1010,750
// DLBL     IJSYSSL,'PRIVATE SOURCE LIB',99/365,SD
// EXTENT   SYSSLB,222222,1,1,1760,230
// DLBL     IJSYS01,'SYS001 WORK FILE',99/365,SD
// EXTENT   SYS001,333333,8,1,0510,250,4
// DLBL     IJSYS02,'SYS002 WORK FILE',99/365,SD
// EXTENT   SYS002,333333,8,1,0515,250,9
// DLBL     IJJSYS03,'SYS003 WORK FILE',99/365,SD
// EXTENT   SYS003,222222,8,1,0510,250,4
// DLBL     IJSYS04,'SYS004 WORK FILE',99/365,SD
// EXTENT   SYS004,222222,8,1,0515,250,9
/*
/&
```

The system, as shown in Figure 8, is now ready for installation of the compiler. Creating partition standard labels before installation enables you to refer to the private libraries during installation without including DLBL and EXTENT statements.

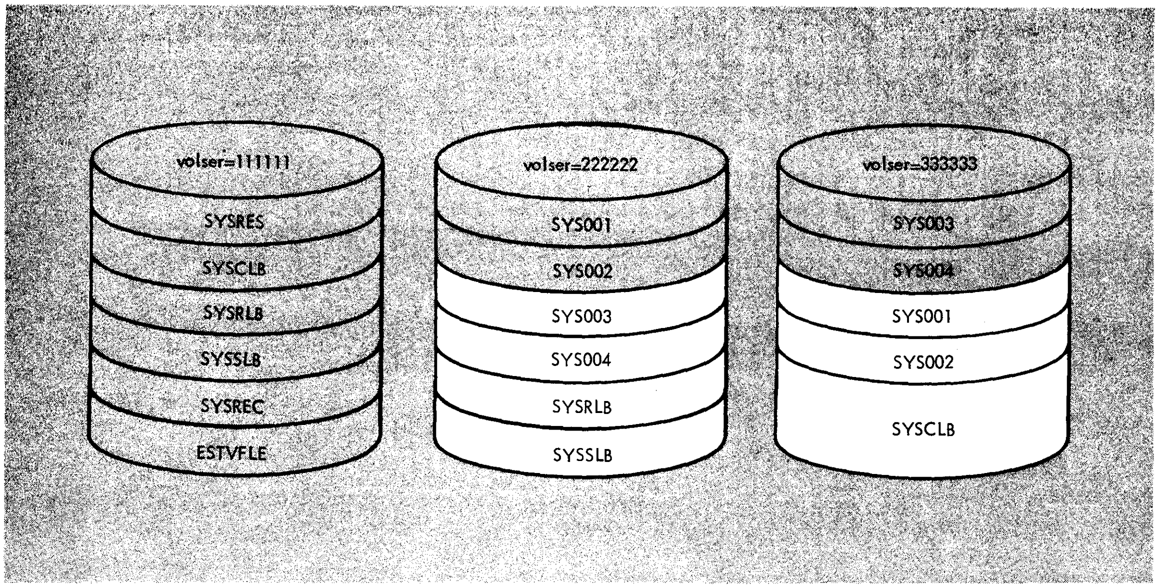

Figure 8. Allocation for Work Files to be Used in Foreground

Notes:

1. A private core image library can be assigned to only one partition at a time. If more than one private core image library is to be used in any one partition, it is advisable not to include the labels in the partition standard labels. A private core image library must always be permanently assigned.

2. Although a private core image library can be assigned in only one partition at any particular time, a relocatable library can be assigned to more than one partition simultaneously.

3. Cataloging (link editing) into a private core-image library is valid in any partition at any time. Cataloging into a private relccatable library, however, is valid only in the background, and only if the library is not assigned to any other partition at that time.

   Because of this restriction in the use of the relocatable library, the first compiler installation job and the first library installation job must be run in the background with a nonshared relocatable library. Subsequent compiler installation jobs and library installation jobs may be run in any partition by assigning the relocatable library used in the first job.

4. The partition priorities are in the following ascending order: background, foreground-1, foreground-2, foreground-3, and foreground-4. These priorities may be changed at installation time through the PRTY parameter of the FOPT macro.

5. If your system contains the relocating loader, it is not necessary to catalog separate copies of the compiler and its object-time subroutines for multipartition compile-link-execute capability. A single copy of the compiler and library residing in the system core image and relocatable libraries will allow simultaneous usage from any partition. The previously discussed rules of library usage still apply, however, and must be considered regarding the use of private libraries during the installation procedure.

6. During installation, it is important to remember that SYSIPT and SYSRDR or SYSIN files cannot be assigned to the same device in two or more partitions at the same time unless they reside on disk. Each partition must have its own input device. Separate output devices must also be assigned for each partition unless they are on disk. For example, if the following assignments are made in the background:

```
ASSGN SYSIPT,X'00C' )
                     } or ASSGN SYSIN,X'00C' (card reader)
ASSGN SYSRDR,X'00C' )
```

then the assignments for a foreground partition must be different, such as those given below to specify tape input.

```
ASSGN SYSIPT,X'182' )
                     } or ASSGN SYSIN,X'182' (tape unit)
ASSGN SYSRDR,X'182' )
```

INSTALLATION FROM TAPE

The compiler and library distribution tape contains 80-byte, fixed-length records, blocked 43 records per block (blocksize is 3440 bytes). The library is also distributed separately on tape, (see "Separate Library Installation from Tape"). The installation tape contains four unlabeled files separated by tape marks.

The first file contains a job which catalogs the compiler and library object modules into the relocatable library. Either the system or a private relocatable library may be used. This job *must* be run in the background partition, and the relocatable library must not be shared during the job.

The second file contains three jobs consisting of the job control statements needed to link edit and catalog the compiler phases and the library transients and dynamically-loaded subroutines into the core image library. Either the system or a private core image library may be used. These jobs may be run in any partition.

The third file is the sample program provided by IBM. Use this program to verify that the compiler and library have been successfully installed. See "Using the Sample Program" in this section for instructions.

The fourth file is for 3886 users. It contains a job which catalogs Data and Procedure Division COPY members into the source statement library. See "3886 Considerations" in this section for instructions.

Installing the Compiler and Library Modules into the Relocatable Library

Before you can install the compiler and library, you must deblock the files on the installation tape. The following example shows how to use the DSTRB utility to deblock the first file, which contains a job to install the compiler and library modules into the relocatable library.

Position the tape at the beginning of the first file and execute the following job.

```
// JOB DEBLOCK
// ASSGN SYS004,X'cuu'
// ASSGN SYS005,X'cuu'
// TLBL UOUT,'file-id'
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu is the unit address of the tape drive on which the tape is mounted. SYS004 is the installation tape supplied by IBM; SYS005 is the tape to be used for deblocked output.

file-id is the unique name for the file on the output tape.

Note: The output tape does not rewind. It must be reset to load point before the compiler and library can be installed. Do not rewind the installation tape.

After the first file has been deblocked and the DOS/VS system libraries have been prepared, use the deblocked tape file to install the compiler and library. Execute the job in the background partition. Assign SYSRDR and SYSIPT to the tape unit on which the deblocked tape is mounted as shown below.

```
ASSGN SYSIPT,X'cuu' }
                    }  or  ASSGN SYSIN,X'cuu'
ASSGN SYSRDR,X'cuu' }
```

where: cuu is the unit address of the tape drive on which the deblocked tape is mounted.

After the operator has entered the command, the system executes the job on the first file. Following the JOB control statement there is a PAUSE control statement. The PAUSE statement enables you to specify a private relocatable library by assigning SYSRLB and SYSCLB at that time.

The PAUSE control statement causes a message to appear on the operator console. To add the compiler and library modules to a private relocatable library, respond to this message by entering an ASSGN command for SYSRLB to define the private library. To add the compiler and library modules to the system relocatable library, simply signal END/ENTER in response to the message. This job must be run in the background, and the relocatable library must not be shared.

Cataloging the Compiler and Library into the Core Image Library

Before you can catalog into the core image library, you must deblock the second file on the installation tape. The following example shows how to use the TPTP utility to deblock the second file.

Position the installation tape at the second file and execute the following job:

```
// JOB DEBLOCK
// ASSGN SYS004,X'cuu'    installation tape
// ASSGN SYS005,X'cuu'    work tape
// TLBL UOUT,'file-id'
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu   is the unit address of the tape drive on which the tape is mounted. SYS004 is the installation tape supplied by IBM; SYS005 is the tape to be used for deblocked output.

file-id is the unique name for the file on the output tape.

Note: The output tape does not rewind. It must be reset to load point before cataloging into the core image library can take place. Do not rewind the installation tape.

After the second file on the installation tape has been deblocked and the DOS/VS system libraries have been prepared, use the deblocked file to catalog into the core image library.

Run this job in the partition in which you wish to install the compiler and library. Assign SYSRDR and SYSIPT to the unit where the deblocked file is ready. Sample commands are:

```
ASSGN SYSIPT,X'cuu' \
                     } or ASSGN SYSIN,X'cuu'
ASSGN SYSRDR,X'cuu' /
```

where: cuu is the unit address of the tape drive or card reader containing the deblocked file.

If you plan to use the compiler and library with CMS/DOS, you must install it in the background partition. You must also link-edit all the compiler relocatable modules using the following linkage editor control statement:

```
ACTION   REL
```

After the operator has entered the command, the system executes the job. Following the job control statement, there is a PAUSE statement.

When the PAUSE card message appears on the console, you can add the compiler and library to a private core image library. To do this, respond to the message by entering an ASSGN command for SYSRLB to define the private relocatable library to which the modules have been added. Enter a second ASSGN command for SYSCLB to define the private core image library to which the compiler and library are to be added. To add the compiler and library to the system core image library, simply signal END/ENTER in response to the message. If you have added the compiler and library to a private relocatable library but want to add them to the system core image library, respond to the message by entering an ASSGN command for SYSRLB to define the private relocatable library to which the compiler and library have been added.

Responses to the PAUSE messages in JOB1 and JOB2 are summarized in Figure 9. The summary assumes the necessary volumes are mounted on disk drives 191 and 192.

PAUSE Message	User Response	Result
JOB1 (File 1)		
IF YOU WISH TO PUT MODULES IN PRIVATE RELOCATABLE LIBRARY, YOU MAY ASSIGN SYSRLB AFTER PAUSE, IF NOT END/ENTER.	ASSGN SYSRLB,X'191'	Compiler and library added to private relocatable library.
	or	or
	END/ENTER	Compiler and library added to system relocatable library.
JOB2 (File 2)		
IF YOU TEMPORARILY ASSIGNED SYSRLB IN THE PRIOR JOB, YOU MUST ASSIGN IT AGAIN. IF YOU DID NOT ASSIGN IT, END/ENTER.	END/ENTER	Compiler and library added to system core image library from system relocatable library.
	or	or
	ASSGN SYSRLB,X'191'	Compiler and library added to system core image library from private relocatable library.
	or	or
	ASSGN SYSCLB,X'192'	Compiler and library added to private core image library from system relocatable library.
	or	or
	ASSGN SYSRLB,X'191' ASSGN SYSCLB,X'192'	Compiler and library added to private core image library from private relocatable library.
Note: Each ASSGN command must be followed by DLBL and EXTENT commands if no label for the private library defined in the command exists in the partition standard labels.		

Figure 9. Responses to PAUSE Messages in Cataloging Compiler and Library From Tape

As a result of executing the two jobs on the first and second files, respectively, of the installation tape, the compiler and library are cataloged into the relocatable library and the core image library that you have specified.

Using the Sample Program

The sample program provided by IBM helps you verify that the compiler has been successfully installed. The sample program, TESTRUN, creates a blocked, labeled, standard sequential file, on a tape assigned to SYS008, and then reads it back in. This tape is assigned to device address 483. A source statement listing appears in the publication DOS/VS COBOL Compiler and Library Programmer's Guide.

To use the sample program, deblock and punch the third file onto cards as follows. Position the installation tape at the beginning of the third file and execute the following job:

```
// JOB SAMPLE
// ASSGN SYS005,X'cuu'        defines the system punch unit
// ASSGN SYS004,X'cuu'        defines the installation tape drive
// EXEC DSTRB
// UDS DBL
// END
/&
```

Note: Do not rewind the installation tape.

Place the card deck produced as output from this job into the card reader. When you are ready to run the sample program, have the operator issue the necessary commands.

Successful compilation of the sample program will produce the following output:

1. A listing of the job control statements.

2. A listing of the source program.

3. A glossary of compiler-generated information about data, as well as global tables and literal pools and register assignments.

4. A listing of the object code.

5. A cross-reference listing.

6. Linkage editor diagnostic output and a map of virtual storage.

7. Execution time output produced by the sample program.

3886 Considerations

The two COBOL source statement library members provided by IBM allow the 3886 user to copy standard Data and Procedure Division statements into a COBOL program. An illustration of these source statements appears in the publication IBM DOS/VS COBOL Compiler and Library Programmer's Guide. Note: The 3886 is not supported in CMS/DOS.

To install these members in the source statement library, deblock and punch the fourth file on the installation tape onto cards as shown below.

```
// JOB DEBLOCK
// ASSGN SYS005,X'cuu'        defines the system punch unit
// ASSGN SYS004,X'cuu'        defines the installation tape drive
// EXEC DSTRB
// UDS DBL
// END
```

Note: Do not rewind the installation tape.

Place the card deck produced as output from this job in the card reader and have the operator issue the necessary commands to execute the job stream.

Generation of Compiler Diagnostic Messages

The user should request a complete listing of the diagnostic messages generated by this compiler. This is done by compiling a program with a PROGRAM-ID of ERRMSG. (In this case, a normal compilation never takes place. Only a list of all error messages and problem determinations is produced. The link option is reset if it was in effect.)

Figure 10 shows a COBOL program that produces compiler diagnostic messages.

```
// JOB    ERRORMSG   User information
//        EXEC       FCOBOL

      ID DIVISION.
      PROGRAM-ID.  ERRMSG.
      REMARKS.  COMPILATION OF THIS PROGRAM WILL RESULT IN ALL
                COMPILER DIAGNOSTICS BEING PRODUCED-NO OBJECT MODULE
                                                    IS PRODUCED.
                                                   -NO COMPILE-TIME
                                                    STATISTICS ARE
                                                    PRODUCED.
      ENVIRONMENT DIVISION.
      DATA DIVISION.
      PROCEDURE DIVISION.
     *          THE SAME RESULTS CAN BE ACHIEVED BY CHANGING THE
     *          PROGRAM-ID OF ANY PROGRAM TO 'ERRMSG'.
              STOP RUN.
```

Figure 10. A Program that Produces COBOL Compiler Diagnostics

SEPARATE LIBRARY INSTALLATION FROM TAPE

The library is also distributed separately on a tape that consists of 80-byte, fixed-length records, blocked 43 records per block (blocksize is 3440 bytes).

The library installation tape contains two unlabeled files separated by tapemarks.

The first file contains a job that catalogs the subroutines and LIOCS modules into the relocatable library. Either the system or a private relocatable library may be used. This job must be run in the background partition, and the relocatable library must not be shared during the job.

The second file contains two jobs that link edit and catalog the library transients and the dynamically-loaded subroutines into the core image library. Either the system or a private core image library may be used. This job may be run in any partition.

Installing the Subroutine Library Modules into the Relocatable Library

Before you can install the library, you must deblock the files on the installation tape. The following example shows how to use the DSTRB utility to deblock the first file.

Position the installation tape at the beginning of the first file and execute the following job.

```
// JOB DEBLOCK
// ASSGN SYS004,X'cuu'
// ASSGN SYS005,X'cuu'
// TLBL UOUT,'file-id'
// EXEC DSTRB
// UDS TLBL
// END
/&
```

where: cuu is the unit address of the tape drive on which the tape is mounted. SYS004 is the installation tape supplied by IBM; SYS005 is the tape to be used for deblocked output.

file-id is the unique name for the file on the output tape.

Note: The output tape does not rewind. It must be reset to load point before the subroutine library modules can be installed. Do not rewind the installation tape.

Use the deblocked file to install the Subroutine Library into the relocatable library. Execute this job in the background partition. Assign SYSRDR and SYSIPT as shown below to the tape unit on which the deblocked tape is mounted.

```
ASSGN SYSIPT,X'cuu' )
                    }  or  ASSGN SYSIN,X'cuu'
ASSGN SYSRDR,X'cuu' )
```

where: cuu is the unit address of the tape drive on which the deblocked tape is mounted.

After the operator has entered the commands, the system executes the job on the first file. Following the JOB control statement, there is a PAUSE control statement. The PAUSE statement enables you to specify a private relocatable library by assigning SYSRLB.

This job adds the subroutine library modules and the LIOCS modules to the relocatable library. It must be run in the background and the relocatable library must not be shared during the job. When the PAUSE message appears, enter an ASSGN command for SYSRLB if you wish the modules to be added to a private relocatable library. To add the library subroutines and LIOCS modules to the system relocatable library, simply signal END/ENTER in response to the message.

Cataloging the Subroutine Library Transients and Dynamically-Loaded Subroutines into the Core Image Library

The following example shows how to use the TPTP utility to deblock the second file. This file contains two jobs to catalog the library transient- and dynamically-loaded subroutines into the core image library.

Position the installation tape at the beginning of the second file and execute the following job.

```
// JOB DEBLOCK
// ASSGN SYS004,X'cuu'    installation tape
// ASSGN SYS005,X'cuu'    work tape
// TLBL UOUT,'file-id'
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu is the unit address of the tape drive on which the tape is mounted. SYS004 is the installation tape supplied by IBM; SYS005 is the tape to be used for deblocked output.

file-id is the unique name of the file on the output tape.

Note: The output tape does not rewind. It must be reset to load point before cataloging of the subroutines into the core image library can be done. Do not rewind the installation tape.

After the second file has been deblocked, and the DOS system libraries have been prepared, use it to catalog the library transients (JOB2) and the dynamically-loaded subroutines (JOB3) into the core image library.

Run this job in the partition in which you wish to install the library. Assign SYSRDR and SYSIPT as shown below to the unit where the deblocked file is ready.

```
ASSGN SYSIPT,X'cuu' )
                    > or ASSGN SYSIN,X'cuu'
ASSGN SYSRDR,X'cuu' )
```

where: cuu is the unit address of the tape drive containing the deblocked file.

If you plan to use the compiler and library with CMS/DOS, you must install it in the background partition. You must also link-edit all the compiler relocatable modules using the following linkage editor control statement:

```
ACTION   REL
```

After the operator has entered the command, the system begins executing the jobs. Following each job control statement, there is a PAUSE control statement.

The first job adds the transient routines to the core image library. When the PAUSE message appears, these routines can be added to a private core image library by entering an ASSGN command for SYSCLB to define the private core image library. To add the transient routines to the system core image library, simply signal END/ENTER in response to the message.

The second job adds the dynamically loaded subroutines to the core image library. These subroutines are self-relocatable. To add the dynamically loaded subroutines to a private core image library, respond to the PAUSE message by entering an ASSGN command for SYSCLB to define the private core image library. To add the routines to the system core image library, simply signal END/ENTER in response to the message.

During link editing of these dynamically loaded subroutines, an unresolved external reference is created for the IOCS module IJJCPDV2. The reference is resolved during execution of the routines.

Responses to the PAUSE messages are summarized in Figure 11. The summary assumes the necessary volumes are mounted on disk drives 191 and 192.

PAUSE Message	User Response	Result
JOB1 (File 1) IF YOU WISH TO PUT SUBROUTINES IN PRIVATE RELOCATABLE LIBRARY, YOU MAY ASSIGN IT AFTER PAUSE.	END/ENTER or ASSGN SYSRLB,X'191'	Subroutines and IOCS modules added to system relocatable library. or Subroutines and IOCS modules added to private relocatable library.
JOB2 (File 2) YOU MAY ASSIGN A PRIVATE CORE IMAGE LIB AT PAUSE. IF NOT, END/ENTER.	END/ENTER or ASSGN SYSCLB,X'192'	Transient routines added to system core image library. or Transient routines added to private core image library.
JOB3 (File 2) YOU MAY ASSIGN A PRIVATE CORE IMAGE LIB AT PAUSE. IF NOT, END/ENTER.	END/ENTER or ASSGN SYSCLB,X'192'	Dynamically loaded subroutines added to system core image library. or Dynamically loaded subroutines added to private core image library.
Note: Each ASSGN command must be followed by DLBL and EXTENT commands if no label for the private library defined in the command exists in the partition standard labels.		

Figure 11. Responses to PAUSE Messages in Cataloging Library From Tape

As a result of executing the three jobs on the first and second files of the installation tape, the library modules are cataloged into the relocatable library and the core image library that you have specified.

The compiler and library can be ordered on a disk supplied by the customer or a disk purchased from IBM.

The installation disk contains four files consisting of 80-byte, fixed-length records, blocked 43 records per block (blocksize is 3440 bytes).

The first file contains a job that catalogs the compiler and library object modules into the relocatable library. Either the system or a private relocatable library may be used. This job must be run in the background partition, and the relocatable library must not be shared during the job. The name of this file on the installation disk is 'N5746CB1.VSANSCOB.JOB1'.

The second file contains three jobs consisting of the control statements needed to link edit and catalog the compiler phases, library transients, and dynamically-loaded subroutines into the core-image library. Either the system or a private core-image library may be used. These jobs may be run in any partition. The name of this file on the installation disk is 'N5746CB1.VSANSCOB.JOB2'.

The third file is the sample program provided by IBM. Use this program to verify that the compiler has been successfully installed. See "Using the Sample Program" in this section for instructions. The name of this file on the installation disk is 'N5746CB1.VSANSCOB.SAMPROG'.

The fourth file is for 3886 users. It contains a job which catalogs Data and Procedure Division COPY statements into the source statement library. See "3886 Considerations" in this section for instructions. The name of this file on the installation disk is 'N5746CB1.VSANSCOB.CPY3886'.

Installing the Compiler and Library Modules into the Relocatable Library

Before you can install the compiler and library, you must deblock the files on the installation disk. The following example shows how to use the DSTRB utility to deblock the first file.

```
// JOB DEBLOCK DISK TO DISK
// ASSGN SYS004,X'cuu'    installation disk
// ASSGN SYS005,X'cuu'    deblocked output disk
// DLBL UIN,'N5746CB1.VSANSCOB.JOB1'
// EXTENT SYS004
// DLBL UOUT,'DOS VS COBOL'
// EXTENT SYS005,111111,1,0,100,500
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu is the unit address of the disk drives. SYS004 is the installation disk supplied by IBM; SYS005 is the disk to be used for the deblocked output.

After the DOS/VS system libraries have been prepared, you may use the deblocked file to install the compiler and library into the relocatable library by running the following job in the background partition.

Place these cards in the card reader and then signal END/ENTER.

```
   // DLBL ILSYSIN,'DOS VS COBOL'


      ASSGN SYSIPT,X'cuu'⎫
                         ⎬ or ASSGN SYSIN,X'cuu'
      ASSGN SYSRDR,X'cuu'⎭
```

where: cuu is the unit address of the drive containing the deblocked file.

After the DOS/VS system libraries have been prepared, have the operator issue the necessary commands to execute the job stream in the background partition.

Following the JOB control statement in the input stream, there is a PAUSE control statement. The PAUSE statement enables you to specify a private relocatable library by assigning SYSRLB.

The PAUSE card causes a message to appear on the console. To add the compiler and library to a private relocatable library, respond to this message by entering an ASSGN command for SYSRLB to define the private library. To add the compiler and library to the system relocatable library, simply signal END/ENTER in response to the message. This job must be run in the background and the relocatable library must not be shared during the job.

Cataloging the Compiler and Library into the Core Image Library

Before you can catalog into the core image library, you must deblock the second file on the installation disk. The following example shows how to use the DSTRB utility to deblock the second file.

```
// JOB DEBLOCK DISK TO DISK
// ASSGN SYS004,X'cuu'    installation disk
// ASSGN SYS005,X'cuu'    deblocked output disk
// DLBL UIN,'N5746CB1.VSANSCOB.JOB2'
// EXTENT SYS004
// DLBL UOUT,'DOS VS COBOL'
// EXTENT SYS005,111111,1,0,600,2
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu is the unit address of the disk drives. SYS004 is the installation disk supplied by IBM; SYS005 is the disk to be used for the deblocked output.

After the DOS system libraries have been prepared, you may use the deblocked file to catalog the compiler and library into the core image library by running the following job in the partition where you would like the compiler and library installed.

Place the following cards in the card reader and signal END/ENTER.

```
  // DLBL IJSYSIN,'DOS VS COBOL'

    ASSGN SYSIPT,X'cuu' )
                        > or ASSGN SYSIN,X'cuu'
    ASSGN SYSRDR,X'cuu' )

  where:  cuu is the unit address of the drive containing the
              deblocked file.
```

After the DOS/VS system libraries have been prepared, have the operator issue the necessary commands to execute the job stream in the partition where you would like the compiler and library installed.

If you plan to use the compiler and library with CMS/DOS, you must install it in the background partition. You must also link-edit all the compiler relocatable modules using the following linkage editor control statement:

```
   ACTION   REL
```

After the operator has initiated the job, a PAUSE card message will appear on the console enabling you to add the compiler and library to a private core image library. To do this, respond to the message by entering an ASSGN command for SYSRLB to define the private <u>relocatable</u> library to which the modules have been added. Enter a <u>second</u> ASSGN command for SYSCLB to define the private <u>core image</u> library to which the compiler and library are to be added. To add the compiler and library to the system core image library, simply signal END/ENTER in response to the message. If you have added the compiler and library to a private relocatable library but want to add them to the system core image library, respond to the message by entering an ASSGN command for SYSRLB to define the private relocatable library to which the compiler and library have been added.

Responses to the PAUSE messages are summarized in Figure 12. The summary assumes the necessary volumes are mounted on disk drives 191 and 192.

PAUSE Message	User Response	Result
JOB1 (File 1) IF YOU WISH TO PUT MODULES IN PRIVATE RELOCATABLE LIBRARY, YOU MAY ASSIGN SYSRLB AFTER PAUSE, IF NOT END/ENTER.	ASSGN SYSRLB,X'191'	Compiler and library added to private relocatable library.
	or END/ENTER	or Compiler and library added to system relocatable library.
JOB2 (File 2) IF YOU TEMPORARILY ASSIGNED SYSRLB IN THE PRIOR JOB, YOU MUST ASSIGN IT AGAIN. IF YOU DID NOT ASSIGN IT, END/ENTER.	END/ENTER	Compiler and library added to system core image library from system relocatable library.
	or ASSGN SYSRLB,X'191'	or Compiler and library added to system core image library from private relocatable library.
	or ASSGN SYSCLB,X'192'	or Compiler and library added to private core image library from system relocatable library.
	or ASSGN SYSRLB,X'191' ASSGN SYSCLB,X'192'	or Compiler and library added to private core image library from private relocatable library.
Note: Each ASSGN command must be followed by DLBL and EXTENT commands if no label for the private library defined in the command exists in the partition standard labels.		

Figure 12. Responses to PAUSE Messages in Cataloging Compiler and Library From Disk

Using the Sample Program

The sample program provided by IBM helps you verify that the DOS/VS compiler has been successfully installed. The sample program, TESTRUN, creates a blocked, labeled, standard sequential file, on a tape assigned to SYS008, and then reads it back in. This tape is assigned to device address 483. A source statement listing appears in the publication DOS/VS COBOL Compiler and Library Programmer's Guide.

To use the sample program to verify the installation, deblock and punch the third file onto cards as shown below.

```
// JOB SAMPLE
// ASSGN SYS006,X'cuu'       defines the system punch unit
// ASSGN SYS009,X'cuu'       defines the installation disk drive
                             (standard labels assumed)
// DLBL UIN,'N5746CB1.VSANSCOB.SAMPROG'
// EXTENT SYS009
// EXEC DSTRB
// UDS DBL
// END
/&
```

Place the card deck produced as output from this job into the card reader. When you are ready to run the sample program, have the operator issue the necessary commands to execute the job stream.

Successful compilation of the sample program will produce the following output:

1. A listing of the job control statements.

2. A listing of the source program.

3. A glossary of compiler-generated information about data, as well as global tables, literal pools, and register assignments.

4. A listing of the object code.

5. A cross-reference listing.

6. Linkage editor diagnostic output and a map of virtual storage.

7. Execution time output produced by the sample program.

3886 Considerations

The two COBOL source statement library members provided by IBM allow the 3886 user to copy standard Data and Procedure Division statements into a COBOL program. An illustration of these source statements appears in the publication IBM DOS/VS COBOL Compiler and Library Programmer's Guide. Note: The 3886 is not supported in CMS/DOS.

To install these members in the source statement library, deblock and punch the fourth file on the installation tape onto cards as shown below.

```
// JOB DEBLOCK
// ASSGN SYS006,X'cuu'       defines the system punch unit
// ASSGN SYS009,X'cuu'       defines the installation disk
                             (standard labels assumed)
// DLBL UIN,'N5746LM4.VSANSLIB.CPY3886'
// EXTENT SYS009
// EXEC DSTRB
// UDS DBL
// END
/&
```

Place the card deck produced as output from this job in the card reader and have the operator issue the necessary commands to execute the job stream.

Generation of Compiler Diagnostic Messages

See "Generation of Compiler Diagnostic Messages" in the section "Installation from Tape."

SEPARATE LIBRARY INSTALLATION FROM DISK

The DOS/VS Subroutine Library can also be ordered separately on a disk supplied by the customer or a disk purchased from IBM.

The installation disk contains two files consisting of 80-byte, fixed-length records, blocked 43 records per block (blocksize is 3440 bytes). These files are shown in Figure 13.

The first file contains the job control statements, control cards, and object modules needed to place the library subroutines and the LIOCS modules required by the compiler into the relocatable library. Either the system relocatable library or a private relocatable library can be used. This job must be run in the background and the relocatable library must not be shared during the job. The name of this file on the installation disk is 'N5746LM4.VSANSLIB.JOB1'.

The second file contains two jobs to link edit and catalog the library transients and dynamically-loaded subroutines into the core image litrary. Either the system core image library or a private core image library can be used. This job may be run in any partition. The name of this file on the installation disk is 'N5746.LM4.VSANSLIB.JOB2'.

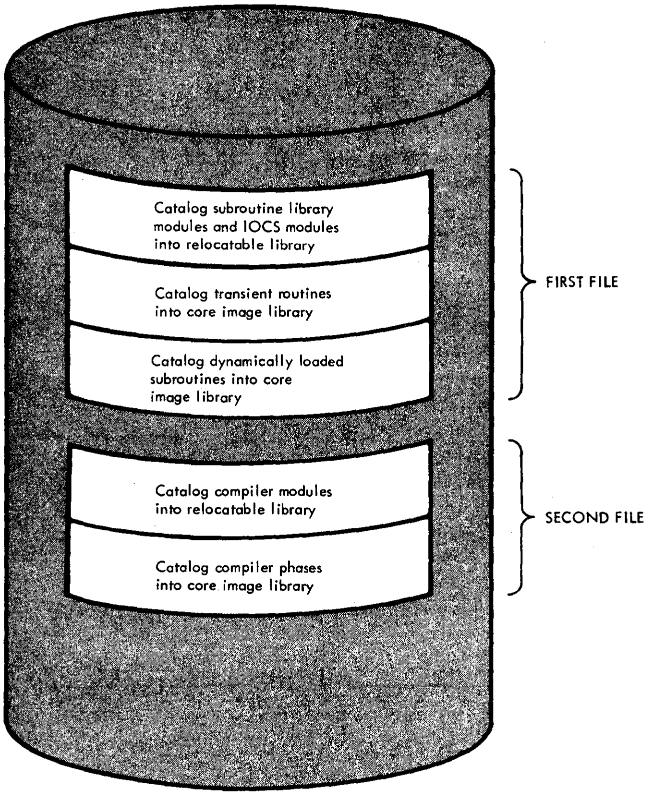
Catalog subroutine library modules and IOCS modules into relocatable library
Catalog transient routines into core image library
Catalog dynamically loaded subroutines into core image library
FIRST FILE
Catalog compiler modules into relocatable library
Catalog compiler phases into core image library
SECOND FILE

Figure 13. Subroutine Library Installation Disk

Installing the Object-Time Subroutines into the Relocatable Library

Before you can install the DOS/VS Subroutine library, you must deblock the files on the installation disk. The following example shows how to use the DSTRB utility to deblock the first file.

```
// JOB DEBLOCK DISK TO DISK
// ASSGN SYS004,X'cuu'    installation disk
// ASSGN SYS005,X'cuu'    deblocked output disk
// DLBL UIN,'N5746LM4.VSANSLIB.JOB1'
// EXTENT SYS004
// DLBL UOUT,'DOS VS COBOL'
// EXTENT SYS005,111111,1,0,100,150
// EXEC DSTRB
// UDD UDS DBL
// END
/&
```

where: cuu is the unit address of the disk drives; SYS004 is the installation disk supplied by IBM; SYS005 is the disk to be used for the deblocked output.

After the DOS system libraries have been prepared, use the deblocked file to install the object-time subroutines into the relocatable library directly from disk by running the following job in the background partition.

Place the following cards in the card reader and signal END/ENTER.

```
//DLBL IJSYSIN,'DOS VS COBOL'

  ASSGN SYSIPT,X'cuu' ⎫
                      ⎬ or ASSGN SYSIN,X'cuu'
  ASSGN SYSRDR,X'cuu' ⎭
```

where: cuu is the unit address of the drive containing the deblocked file.

Have the operator issue the necessary commands to execute the job stream. The job stream must be run in the background partition.

Following the JOB control statement in the input stream, there is a PAUSE control statement. The PAUSE statement enables you to specify a private relocatable library assigning SYSRLB.

This job adds the subroutine library modules and the IOCS modules to the relocatable library.. When the PAUSE message appears, enter an ASSGN command for SYSRLB if you wish the modules to be added to a private relocatable library. To add the library modules and IOCS subroutines to the system relocatable library, simply signal END/ENTER in response to the message. This job must be run in the background, and the relocatable library must not be shared during the job.

Cataloging the Library Transients and Dynamically-Loaded Subroutines into the Core Image Library

The follcwing example shows how to use the DSTRB utility to deblock the second file, which contains two jobs to catalog the library transients and dynamically-loaded subroutines into the core image library.

```
// JOB DEBLOCK DISK TO DISK
// ASSGN SYS004,X'cuu'    installation disk
// ASSGN SYS005,X'cuu'    deblocked output disk
// DLBL UIN,'N5746LM4.VSANSLIB.JOB2'
// EXTENT SYS004
// DLBL UOUT,'DOS VS COBOL'
// EXTENT SYS005,111111,1,0,250,100
// EXEC DSTRB
// UDS DBL
// END
/&
```

where: cuu is the unit address of the disk drives; SYS004 is the installation disk supplied by IBM; SYS005 is the disk used for the deblocked output.

After the DOS/VS system libraries have been prepared, you may use the deblocked file to catalog the library transients and dynamically-loaded subroutines into the core image library directly from disk by running the following job in the partition where you would like the library installed.

Place the following cards in the card reader and signal END/ENTER.

```
// DLBL IJSYSIN,'DOS VS COBOL'

  ASSGN SYSIPT,X'cuu' )
                      } or ASSGN SYSIN,X'cuu'
  ASSGN SYSRDR,X'cuu' )
```

where: cuu is the unit address of the drive containing the deblocked file.

If disk space is severely limited, you may want to install the library transients and dynamically-loaded subroutines from card input (there are approximately 800 cards). First use the following job control statements to punch the installation jobs from the disk onto cards.

```
// JOB DSKTOCD
// ASSGN SYS006,X'cuu'    defines system punch unit
// ASSGN SYS009,X'cuu'    defines installation disk drive
                          (standard labels are assumed)
// DLBL UIN,'N5746LM4.VSANSLIB.JOB2'
// EXTENT SYS009
// EXEC DSTRB
// UDS DBL
// END
/&
```

When the job is completed, place the output deck into the card reader. Have the operator issue the necessary commands to execute the job stream. The job stream must be run in the partition in which the library transients and dynamically-loaded subroutines are to be installed.

If you plan to use the compiler and library with CMS/DOS, you must install it in the background partition. You must also link-edit all the compiler relocatable modules using the following linkage editor control statement:

```
ACTION   REL
```

After the operator has entered the command and indicated END/ENTER, the system begins executing the jobs. Following each JOB control statement there is a PAUSE control statement.

The first job adds the transient routines to the core image library. When the PAUSE message appears, these routines can be added to a private core image library by entering an ASSGN command for SYSCLB to define the private core image library. To add the transient routines to the system core image library, simply signal END/ENTER in response to the message.

The second job adds the dynamically loaded subroutines to the core image library. These subroutines are self-relocatable. To add the dynamically loaded subroutines to a private core image library, respond to the PAUSE message by entering an ASSGN command for SYSCLB to define the private core image library. To add the routines to the system core image library, simply signal END/ENTER in response to the message.

During link editing of these dynamically loaded subroutines, an unresolved external reference is created for the IOCS module IJJCPDV2. The reference is resolved during execution of the routines.

Responses to the PAUSE messages are summarized in Figure 14. The summary assumes the necessary volumes are mounted on disk drives 191 and 192.

PAUSE Message	User Response	Result
JOB1 (File 1) IF YOU WISH TO PUT SUBROUTINES IN PRIVATE RELOCATABLE LIBRARY, YOU MAY ASSIGN IT AFTER PAUSE.	END/ENTER or ASSGN SYSRLB,X'191'	Subroutines and IOCS modules added to system relocatable library. or Subroutines and IOCS modules added to private relocatable library.
JOB2 (File 2) YOU MAY ASSIGN A PRIVATE CORE IMAGE LIB AT PAUSE. IF NOT, END/ENTER.	END/ENTER or ASSGN SYSCLB,X'192'	Transient routines added to system core image library. or Transient routines added to private core image library.
JOB3 (File 2) YOU MAY ASSIGN A PRIVATE CORE IMAGE LIB AT PAUSE. IF NOT, END/ENTER.	END/ENTER or ASSGN SYSCLB,X'192'	Dynamically loaded subroutines added to system core image library. or Dynamically loaded subroutines added to private core image library.

Figure 14. Responses to PAUSE Messages in Cataloging Library From Disk

Note: Each ASSGN command must be followed by DLBL and EXTENT commands if no label for the private library defined in the command exists in the partition standard labels.

STORAGE REQUIREMENTS

This section defines the partition size required by the DOS/VS compiler alone and the storage required by the compiler together with the DOS/VS subroutine library on the core image library. The storage needed for the compiler, subroutines, and required IOCS modules on the relocatable library is also given.

VIRTUAL STORAGE

Minimum 60K bytes of virtual storage. The compiler needs added virtual storage if a BUF parameter is specified on the CBL card. Enough extra storage must be allocated to compensate for six buffers of that size.

CORE IMAGE LIBRARY

Number of Phases		Number of Library Blocks For Release 29* (Physical Records)
Compiler	24	511
Transient routines	4	4
Library Phases	14	32

*All supported devices.

Compiler Phases

FCOBOL	
FCOBOL05	
FCOBOL06	(Note 1)
FCOBOL07	
FCOBOL08	
FCOBOL10	
FCOBOL11	
FCOBOL12	
FCOBOL20	
FCOBOL21	
FCOBOL22	
FCOBOL25	(Note 2)
FCOBOL30	
FCOBOL40	
FCOBOL50	
FCOBOL51	
FCOBOL60	(Note 3)
FCOBOL62	
FCOBOL63	
FCOBOL64	
FCOBOL65	(Note 2)
FCOBOL61	
FCOBOL70	
FCOBOL80	(Note 4)

Note 1: Phases FCOBOL05, FCOBOL06, FCOBOL07, and FCOBOL08 are executed only when the lister feature is used.

Note 2: Phases FCOBOL25 and FCOBOL65 are executed only when symbolic debugging is used.

Note 3: If optimization is not requested, phase FCOBOL60 is called instead of FCOBOL62, FCOBOL63, and FCOBOL64.

Note 4: If FIPS processing is not requested, phase FCOBOL80 is not called, and compilation terminates after phase FCOBOL70.

Transient Routines

$$BCOBEM (Executed only when SYMDMP, FLOW, STATE, or COUNT is specified.)
$$BCOBER
$$BCOBR1
$$BFCMUL

Library Phases

Executed only when SYMDMP is specified.

ILBDMP01
ILBDMP02
ILBDMP04
ILBDMP10
ILBDMP11
ILBDMP12
ILBDMP13
ILBDMP14
ILBDMP20
ILBDMP21
ILBDMP22
ILBDMP23
ILBDMP24
ILBDMP25

RELOCATABLE LIBRARY

Number of Modules		Number of Library Blocks (Physical Records)
Compiler	39	2473
Library	81	425
IOCS	7	32

Compiler Modules

ILACBVS
ILACBL00
ILACBL01
ILACBL05
ILACBL06
ILACBL07
ILACBL08
ILACBL10
ILACBL11

ILACBL12
ILACBL20
ILACBL21
ILACBL22
ILACBL25

ILACBL30
ILACBL40
ILACBL50
ILACBL51
ILACBL60

ILACBL62
ILACBL63
ILACBL64
ILACBL65
ILACBL61
ILACBL70

ILACBL80
ILACBL81
ILACBL82
ILACBL83
ILACBL84
ILACBL85
ILACBL86
ILACBL87
ILACBL88
ILACBL89
ILACBL8A
ILACBL8B
ILACBL8C
ILACBL8D

Subroutines

Library Modules	Bytes of Virtual Storage	Blocks
ILBDABX0	412	4
ILBDACP0	1050	7
ILBDADR0	350	4
ILBDANE0	322	3
ILBDANF0	110	2
ILBDASY0	88	2
ILBDATB0	256	2
ILBDBID0	112	2
ILBDBIE0	116	2
ILBDBII0	464	4
ILBDCKP0	849	6
ILBDCLK0	56	2
ILBDCLS0	150	2
ILBDCRD0	150	2
ILBDCT10	480	3
ILBDDAE0	344	4
ILBDDBG0	2804	16
ILBDDCI0	177	5
ILBDDIO0	720	5
ILBDDSP0	1328	9
ILBDDSR0	334	4
ILBDDSS0	808	7
ILBDDUM0	2	2
ILBDEFL0	524	6
ILBDETB0	256	4
ILBDFLW0	1256	9
ILBDFMT0	182	3
ILBDFPW0	808	5
ILBDGDO0	160	2
ILBDGPW0	88	2
ILBDIDA0	396	3
ILBDIDB0	116	2
ILBDIDR0	1663	8
ILBDIDT0	692	4
ILBDIFB0	296	3
ILBDIFD0	156	5
ILBDIML0	88	2
ILBDINT0	288	4
ILBDISE0	480	4
ILBDISM0	366	3
ILBDITB0	256	12
ILBDIVL0	72	2
ILBDMFT0	148	3
ILBDMNS0	1	2
ILBDMOV0	66	3
ILBDMRG0	1148	8
ILBDMVE0	224	3
ILBDNSL0	616	5
ILBDOCR0	1922	10
ILBDOSY0	132	2
ILBDRCR0	148	2
ILBDRDI0	408	4
ILBDRDS0	238	3

Library Modules	Bytes of Virtual Storage	Blocks
ILBDRFM0	132	2
ILBDSAE0	400	4
ILBDSCH0	744	4
ILBDSEM0	279	5
ILBDSET0	296	3
ILBDSMV0	60	2
ILBDSPA0	1524	8
ILBDSRT0	3720	18
ILBDSTR0	80	3
ILBDSSN0	300	3
ILBDSTI0	596	4
ILBDSTN0	1736	8
ILBDTC00	1168	7
ILBDTC20	500	3
ILBDTC30	4217	19
ILBDTEF0	600	6
ILBDTOD0	200	2
ILBDTRN0	256	2
ILBDUPS0	108	2
ILBDUSL0	365	3
ILBDVBL0	314	4
ILBDVCO0	512	3
ILBDVIO0	4658	20
ILBDVMO0	468	4
ILBDVOC0	4276	18
ILBDVTR0	138	2
ILBDWTB0	256	3
ILBDXDI0	273	3
ILBDXMU0	184	2
ILBDXPR0	608	6
ILBDXTN0	288	2

IOCS Modules

Needed when the ASCII Support Feature is used.

IJFXBZZN
IJFXBZZZ
IJFXZZZZ
IJFNZZZN
IJFNZZZZ
IJFRZZZN
IJFRZZZZ

This section describes diagnostic messages issued in connection with the DOS/VS Compiler and Subroutine Library. They are grouped as follows:

- Compile-time messages from the compiler concerning the compilation of the source program, e.g., syntax.
- Operator messages from the compiler concerning conditions that inhibit its functioning, e.g., too small a partition.
- Object-time messages issued by the object program itself that are numbered.
- Object-time messages that are not numbered.

This section should be used with the publication DOS/VS Messages.

COMPILE-TIME MESSAGES

A complete listing of all the error messages produced by the compiler can be obtained by compiling a program with a PROGRAM-ID of ERRMSG. The listing consists of:

- Compiler identification and level
- Brief description of the remainder of the listing
- Description of listing conventions
- Action codes and meanings
- General procedure in recurring problem situations
- Error messages in the following format:
  - -- Message number
  - -- Severity level
  - -- Action code
  - -- Message text

OPERATOR MESSAGES

The following messages are issued during compilation on SYSLOG. They are also printed on SYSLST with the prefix ILA.

C100I PARTITION IS LESS THAN 60K.

Explanation: At least 60K is required to compile using DOS/VS COBOL. Probable user error.

System Action: The compilation is terminated.

Operator Response: Make sure the compiler has at least 60K available to it. If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support:

1. Execute the MAP command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C101I DEVICE NOT ASSIGNED - SYSnnn.

Explanation: (nnn is either 001, 002, 003, or 004.) The specified logical unit is unassigned and must be assigned. Probable user error.

System Action: The compilation is terminated.

Operator Response: Use the ASSGN command to assign a physical unit (magnetic tape or disk) to the file indicated. If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support:

1. Execute the LISTIO command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C102I UNSUPPORTED DEVICE TYPE - SYSnnn.

Explanation: (nnn is either 001, 002, 003, or 004.) The specified file must be a tape or disk file for SYS002 through SYS004. SYS001 should be assigned to disk; however, in small, simple programs that do not require dictionary spill, it is sometimes possible to compile with the spill file (SYS001) assigned to tape. If any spill does occur, an input/output error may occur. Compile-time statistics will say "DICTIONARY SPILL HAS OCCURRED". No mention is made of dictionary spill in the compile-time statistics if spill does not occur. Probable user error.

System Action: The compilation is terminated.

Operator Response: Use the ASSGN command to assign the appropriate physical unit to the file indicated -- SYS001 should be assigned to a magnetic tape or disk unit. If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support:

1. Execute the LISTIO command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C103I END OF FILE ON SYSIPT.

Explanation: End-of-file was encountered in the initialization phase; no source statements were found. Probable user error.

System Action: The compilation is terminated.

Operator Response: Ensure that a /* card does not precede the source deck, or add the source deck to the job stream. If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support:

1. Execute the LISTIO command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C104I SYS001 FILE NOT ASSIGNED TO DISK

Explanation: In small, simple programs that do not require dictionary spill, it is sometimes possible to compile with the spill file (SYS001) assigned to tape. However, if any spill does occur, an input/output error may occur. Any compilation which spills the dictionary will contain a message in compile-time statistics. User error.

System Action: The compilation continues.

Operator Response: Use the ASSGN command to assign SYS001 to a disk unit. If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support:

1. Execute the LISTIO command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C105I W-CANNOT OPEN SYS005 -- SYMDMP IGNORED.

Explanation: The SYMDMP option has been specified, but the file needed for symbolic debug cannot be opened since SYS005 is unassigned. Probable user error.

System Action: The SYSDMP option is canceled, and the compilation continues.

Operator Response: Use the ASSGN command to assign SYS005 to a physical unit. If the problem recurs, do the following to complete your problem determination before calling IBM for programming support:

1. Execute the LISTIO command and save the output.

2. Have the source deck, control cards, output listing, and console sheet available.

C106I SYS006 IS NOT A DISK. NOLVL ASSUMED.

Explanation: The specified logical unit is not assigned to a disk.

System Action: Compilation continues with NOLVL.

Operator Response: Use the ASSGN command to assign SYS006 to a disk unit.

NUMBERED OBJECT-TIME MESSAGES

The following messages are normally issued on SYSLOG.

C110A STOP literal

Explanation: The programmer has issued a STOP literal statement in the DOS/VS COBOL source program.

System Action: Awaits operator response.

Programmer Response: Not applicable.

Operator Response: Respond with end-of-block, or with any character in order to proceed with the program.

C111A AWAITING REPLY

Explanation: This message is issued in connection with the DCS/VS COBOL ACCEPT statement.

System Action: Awaits operator response.

Programmer Response: Provide the operator with instructions.

Operator Response: Reply as specified by the programmer.

The following messages are issued on SYSLOG and SYSLST prior to cancellation of the job. If the DUMP option is specified, a partial dump is taken from the problem program origin to the highest virtual location of the last phase loaded. When this occurs, the eight bytes immediately preceding the DTF are destroyed. The messages have the form:

```
CmmmI SYSnnn filename DTFaddress text
```

where:

mmm and text correspond as follows:

mmm	text
112	DATA CHECK
113	WRONG LENGTH RECORD
114	PRIME DATA AREA FULL
115	CYLINDER INDEX TOO SMALL
116	MASTER INDEX TOO SMALL
117	OVERFLOW AREA FULL
118	DATA CHECK IN COUNT
119	DATA CHECK IN KEY OR DATA
120	NO ROOM FOUND
121	DASD ERROR
122	DASD ERROR WHILE ATTEMPTING TO WRITE RECORD ZERO
123	FILE CANNOT BE OPENED AFTER CLOSE WITH LOCK
124	CYLINDER AND MASTER INDEX TOO SMALL

nnn is equal to 001 through 255

filename is seven or fewer characters and is generated from the file-name specified in the SELECT sentence.

address is the hexadecimal address of the file's DTF table

Explanation: Condition indicated occurred on SYSnnn.

System Action: The job is canceled.

Programmer Response: Rerun the job or add a user Declarative Section to the Procedure Division of the source program to handle errors within the program.

If the recurs, do the following problem before calling IBM for programming support: have source deck, control cards, compiler output, and console sheet available.

Operator Response: Not applicable.

C125I NO EXTENTS

Explanation: During CLOSE UNIT processing, no extent is found for the next volume.

System Action: The job is canceled.

Programmer Response: Rerun job with proper EXTENT (XTENT) statements.

If the problem recurs, do the following to complete your problem determination action before calling IBM for programming support: have source deck, control cards, compiler output, and console sheet available.

Operator Response: Not applicable.

The following message is issued on SYSLOG:

C126D SYSnnn IS IT EOF?

where nnn is equal to 001 through 255

Explanation: A tapemark was just read on an unlabeled tape file described at compilation time as having more than one reel.

System Action: Awaits response from operator.

Programmer Response: Not applicable.

Operator Response: Respond with N if it is end of volume, or with Y if it is end of file.

The following messages are issued on SYSLOG and SYSLST:

C127D NO EOF RECORD WRITTEN IN PRIME DATA AREA

Explanation: During CLOSE processing of an ISAM file opened OUTPUT, no room was found to write an EOF record.

Programmer Response: Rerun the job with the proper EXTENT.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, compiler output, and console sheet available.

Operator Response: Not applicable.

C128D UNRECOVERABLE I/O ERROR

Explanation: This is probably a hardware error on tape.

Programmer Response: Not applicable.

Operator Response: Rerun the job.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, compiler output, and console sheet available.

C130I INPUT/OUTPUT ERROR. FILE STATUS SET TO xx NEAR REL LOC. xxxxxx.

Explanation: An I/O error has occurred on the file being accessed by the COBOL statement at or near the relative location given in the message, and the user has no USE declarative for that file.

Action: Control returns to COBOL at the statement following the COBOL request that caused the error.

Programmer Response: If the error occurred on a READ operation, processing can continue. If the error occurred on a WRITE operation, there may be a loss of data.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C131I UNABLE TO OPEN FILE 'SYSnnn'. CANCELING.

Explanation: The VSAM OPEN or CLOSE request gave a return code of X'68' or X'6C' because of invalid time stamps in the VSAM catalog or VSAM file. The VSAM catalog or file should be recreated. See DOS/VS Supervisor and I/O Macros for more detail on the OPEN/CLOSE return codes.

Action: The job is canceled.

Programmer Response: Recreate the VSAM catalog and/or file.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C140I INVALID SEPARATE SIGN CONFIGURATION

Explanation: During execution of a COBOL program, an invalid sign was detected for a separately signed item.

Action: The job is terminated.

Programmer Response: Probable user error. Correct program's input data before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, compiler output, and data available.

The following messages (C150I-C175I) are listed on SYSLST. The messages have the form:

$$\text{CmmmI} \left\{ \begin{array}{l} \text{program-id} \\ \\ \text{card/verb number} \end{array} \right\} \text{text}$$

Messages C150I through C162I may appear interspersed among the SYMDMP control cards at the point at which the error is encountered. Program-id is provided for all messages except C150I through C152I. For these, the card/verb number of the corresponding line-control card is given instead. The program-id associated with C150I through C152I can be determined from the nearest preceding program-control card.

Messages C153I through C155I may also appear in the midst of the dump output if the error condition is not recognized until dumping has started.

Message C171I may appear interspersed with the SYSLST output. It is preceded by the program-id for the program affected.

Message C172I may appear just before the start of execution statistics printing. It applies to the entire run unit.

C150I IDENTIFIER NOT FOUND.

Explanation: An identifier specified on the line-control card cannot be found in the program or is invalid. Level-66 and level-88 items and items defined under an RD are invalid requests.

Action: The dump request for this identifier is ignored.

Programmer Response: Probable user error. Before reexecuting, ensure that no requests have been made on the line-control card for the dumping of identifiers that have not been defined or that are invalid.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C151I CARD NUMBER NOT FOUND

Explanation: The card number specified on the line-control card is not within range of the Procedure Division.

Action: The line-control card which specifies the nonexistent card number is skipped.

Programmer Response: Probable user error. Ensure that any card number specified on a line-control card is within range of numbers specified for source program before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have the source deck, control cards, and compiler output available.

C152I VERB NUMBER NOT FOUND

Explanation: The verb number specified on a line-control card does not exist on the card specified.

Action: The nearest verb number on the card specified is used.

Programmer Response: Probable user error. Correct verb number specification before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C153I NC ROOM TO DUMP.

Explanation: If this message immediately follows a program-control card, sufficient storage is not available for the debug subroutine or for the 72 bytes of data required for each program in the run unit. If this message follows an abnormal termination message, one or more of the following is not available in free storage or in the COBOL Procedure Division: a contiguous block of 4000 bytes, a contiguous block of 1800 bytes, or a contiguous block of 512 bytes.

Action: No Data Division dump for the indicated program and, in some instances, no statement number information, is provided.

Programmer Response: Probable user error. Increase the size of the partition before reexecuting. See "Symbolic Debug Considerations" for information about storage requirements for symbolic debugging.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C154I I/O ERROR ON DEBUG FILE.

Explanation: An input/output error has occurred on the debug file. Note that such an error may be the result of a file other than the debug file being mounted on the logical unit specified.

Action: SYMDMP output is cancelled for the indicated program.

Response: Hardware, operator, or user JCL error. Before reexecuting, check logical unit number specified on program-control card against current mounting, as well as the ASSGN, DLBL, and EXTENT cards of compilation.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C155I WRONG DEBUG FILE FOR PROGRAM.

Explanation: The file corresponding to the filename and/or logical unit number provided on the program-control card is not the debug file created for this program at compile time.

Action: SYMDMP output is cancelled for the indicated program.

Programmer Response: Probable user error. Before reexecuting, ensure that the filename and/or logical unit specified on the program-control card corresponds to that of the debug file created at compile time.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C156I NC ROOM FOR DYNAMIC DUMP.

Explanation: Sufficient storage is not available to store the line-control card information during execution.

Action: Dynamic dumping is cancelled for the indicated program.

Program Response: Probable user error. Increase size of partition or decrease number of line-control cards before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C157I INVALID FILENAME.

Explanation: If the "filename" parameter is specified for a disk file on the CBL card at compile time, the same "filename" must also be specified on the program-control card. "Filename" may be from one to seven characters in length; the first character must be a letter.

Action: All SYMDMP output is cancelled for the indicated program.

Programmer Response: Probable user error. Correct "filename" specification on the program-control card before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C158I INVALID LOGICAL UNIT.

Explanation: The logical unit parameter on the program-control card must be specified, must be an integer between 0 and 244, and must match the one specified in the ASSGN control statement for the debug at compile time.

Action: All SYMDMP output is cancelled for the indicated program.

Programmer Response: Probable user error. Correct logical unit specification on program-control card before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C159I MISSING PARAMETERS.

Explanation: A non-continued line-control card ends with HEX, OF, IN, or THRU. Possibly a continuation punch is missing in column 72.

Action: A HEX or THRU option ending a card is ignored. When a card ends with OF or IN, the word is ignored and the identifier that is dumped is the first one encountered whose qualifiers match those preceding the word OF or IN.

Programmer Response: Probable user error. Check line-control card for keypunch errors before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C160I INVALID OPTION.

Explanation: An element used as an optional parameter on a program-control card is not one of the legal program-control card options.

Action: The element is ignored.

Programmer Response: Probable user error. Correct syntax of program-control card before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C161I SUBSCRIPTING ILLEGAL.

Explanation: The "name" parameter of the line-control card may nct be subscripted.

Action: The subscripts are ignored. Every occurrence of the identifier is dumped.

Programmer Response: Probable user error. Specify the name of the item without the subscripts before reexecuting. This will result in a dump of every occurrence of the item.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C162I ON PARAMETER TOO BIG.

Explanation: Neither the n, m, nor k parameter of the ON option may exceed 32767.

Action: The number is reduced to 32767.

Programmer Response: Probable user error. Correct invalid parameter before reexecuting.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C163I FLOW TRACE NON-CONTIGUOUS. MORE THAN 10 PROGRAMS ENCOUNTERED

Explanation: A non-contiguous flow trace will result if FLOW option is effective in a subprogram structure of more than 10 programs compiled with the FLOW option.

Action: The FLOW is terminated upon encountering the eleventh PROGRAM-ID. Tracing resumes only upon returning to one of the original tem programs.

Programmer Response: Probable user error. If trace is absent for a program where it is critical, recompile one or more of the programs where the flow is non-critical without the FLOW option and reexecute.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C164I FLOW TRACE IN EFFECT BUT NO PROCEDURES TRACED.

Explanation: Abnormal termination has taken place before any COBOL statement with a procedure-name has been traced.

Action: No tracing is done.

Programmer Response: Probable user error. If trace is desired, recompile the program after inserting additional procedure-names.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C165I SYMDMP/STATE/FLOW/COUNT INTERNAL ERROR. EXECUTION CANCELLED.

Explanation: Abnormal termination occurred during execution of one of the debugging subroutines.

Action: The job is cancelled.

Programmer Response: Internal logic error.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C169I STATE OPTION CANCELLED.

Explanation: Compiler or logic error has occurred during STATE option processing. Under certain conditions, this error may result from other user errors. For example, a loop might destroy some of the information required by the STATE subroutines; an invalid branch might cause a non-existent priority-number to be stored in the TGT, etc.

Action: STATE output is cancelled.

Programmer Response: Probable user error. Possible compiler error or user error. Correct other known errors (if any) before attempting reexecution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C170I INVALID ADDRESS.

Explanation: The address calculated for a subscripted identifier, or a starting or ending address of a variable-length identifier used as the receiving field in a MOVE statement is invalid.

Action: A symbolic dump is produced.

Programmer Response: Probable user error. Possible compiler error or user error. Correct other known errors (if any) before attempting reexecution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C171I SPACE NOT FOUND FOR THE COUNT CHAIN. CONTINUING.

Explanation: A GETVIS macro was unsuccessful due to lack of space.

Action: Execution continues without providing execution statistics for the last indicated program unit.

Programmer Response: Probable user error. Allocate more space on EXEC card before attempting reexecution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C172I SPACE NOT FOUND FOR THE VERBSUM TABLE. CONTINUING.

Explanation: A GETVIS macro was unsuccessful due to lack of space.

Action: Execution continues without providing verb summary statistics for the program.

Programmer Response: Probable user error. Allocate more space on EXEC card before attempting reexecution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C173I FREEVIS FAILED. EXECUTION CANCELLED.

Explanation: A FREEVIS macro was unsuccessful.

Action: Execution is terminated.

Programmer Response: Probable user error. Allocate more space on EXEC card before attempting reexecution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

C175I INVALID COUNT TABLE ENTRY. EXECUTION CANCELLED.

Explanation: A count table entry in the object module is not one of the following: end-of-table indicator, procedure-id, or verb-id.

Action: Execution is terminated.

Programmer Response: Probable user error. Possible compiler or user error. Check your program for routines that may have moved data into the count table area. Correct other known errors (if any) before attempting execution.

If the problem recurs, do the following before calling IBM for programming support: have source deck, control cards, and compiler output available.

UNNUMBERED OBJECT-TIME MESSAGES

xxx...

Explanation: This message is written on the console and is recognizable because it is not preceded by a message code and action indicator. It is issued by an object program originally coded in COBOL. The message text is supplied by the object program and may indicate alternative action to be taken.

System Action: The job continues.

Operator Response: Operator response, if any is needed, is determined by the message text.

IBM DOS/VS COBOL Compiler and
Library Installation Reference Material
SC28-6479-2

Reader's Comment Form

Your comments about this publication will help us to improve it for you. Comment in the space below, giving specific page and paragraph references whenever possible. All comments become the property of IBM.

Please do not use this form to ask technical questions about IBM systems and programs or to request copies of publications. Rather, direct such questions or requests to your local IBM representative.

If you would like a reply, please provide your name, job title, and business address (including ZIP code).

Fold on two lines, staple, and mail. No postage necessary if mailed in the U.S.A. (Elsewhere, any IBM representative will be happy to forward your comments.) Thank you for your cooperation.

SC28-6479-2

Fold and Staple

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE:

IBM Corporation
General Products Division
Programming Publishing—Department J57
1501 California Avenue
Palo Alto, California 94304

Fold and Staple

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601

IBM DOS/VS COBOL IRM Printed in U.S.A. SC28-6479-2

SC28-6479-2

IBM DOS/VS COBOL IRM Printed in U.S.A. SC28-6479-2

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, N.Y. 10604

IBM World Trade Americas/Far East Corporation
Town of Mount Pleasant, Route 9, North Tarrytown, N.Y., U.S.A. 10591

IBM World Trade Europe/Middle East/Africa Corporation
360 Hamilton Avenue, White Plains, N.Y., U.S.A. 10601