SERIES 60 (LEVEL 6)

GCOS 6 MOD 400 PROGRAMMER'S GUIDE ADDENDUM A

SUBJECT

Changes and Additions to the Manual

SPECIAL INSTRUCTIONS

Insert attached pages into the manual (Revision 0, dated January 1978) according to the collating instructions on the back of this cover. Except in the completely revised Section 5, change bars indicate new and changed information and asterisks denote deletions.

Note:

Insert this cover behind the manual cover to indicate that the manual is updated with this addendum.

SOFTWARE SUPPORTED

This update supports Release 0110 of the Series 60 (Level 6) GCOS 6 MOD 400 software system. See the Manual Directory of the *System Concepts* manual regarding later releases supported by this manual.

ORDER NUMBER

CB22A, Rev. 0

June 1978

20967
3678
Printed in U.S.A.

Honeywell

Collating Instructions

To update this manual, remove old pages and insert new pages as follows:

Remove	*Insert*
iii, blank	iii, blank
v, vi	v,vi
1-1, 1-2	1-1, 1-2
1-5, blank	1-5, blank
2-1, 2-2	2-1, 2-2
5-1 through 5-14	5-1 through 5-14
5-15, blank	5-15, blank
7-3 through 7-8	7-3 through 7-8

File No.: 1S23

6/78
CB22A

MANUAL DIRECTORY

The following publications comprise the GCOS 6 manual set. The Manual Directory in the latest *GCOS 6 MOD 400 Systems Concepts* manual (Order No. CB20) lists the current revision number and addenda (if any) for each manual in the set.

Order No.	*Manual Title*
CB01	*GCOS 6 Program Preparation*
CB02	*GCOS 6 Commands*
CB03	*GCOS 6 Communications Processing*
CB04	*GCOS 6 Sort/Merge*
CB05	*GCOS 6 Data File Organizations and Formats*
CB06	*GCOS 6 System Messages*
CB07	*GCOS 6 Assembly Language Reference*
CB08	*GCOS 6 System Service Macro Calls*
CB09	*GCOS 6 RPG Reference*
CB10	*GCOS 6 Intermediate COBOL Reference*
CB20	*GCOS 6 MOD 400 System Concepts*
CB21	*GCOS 6 MOD 400 Program Execution and Checkout*
CB22	*GCOS 6 MOD 400 Programmer's Guide*
CB23	*GCOS 6 MOD 400 System Building*
CB24	*GCOS 6 MOD 400 Operator's Guide*
CB25	*GCOS 6 MOD 400 FORTRAN Reference*
CB26	*GCOS 6 MOD 400 Entry-Level COBOL Reference*
CB27	*GCOS 6 MOD 400 Programmer's Pocket Guide*
CB28	*GCOS 6 MOD 400 Master Index*
CB30	*Remote Batch Facility User's Guide*
CB31	*Data Entry Facility User's Guide*
CB32	*Data Entry Facility Operator's Quick Reference Guide*
CB33	*Level 6/Level 6 File Transmission Facility User's Guide*
CB34	*Level 6/Level 62 File Transmission Facility User's Guide*
CB35	*Level 6/Level 64 (Native) File Transmission Facility User's Guide*
CB36	*Level 6/Level 66 File Transmission Facility User's Guide*
CB37	*Level 6/Series 200/2000 File Transmission Facility User's Guide*
CB38	*Level 6/BSC 2780/3780 File Transmission Facility User's Guide*
CB39	*Level 6/Level 64 (Emulator) File Transmission Facility User's Guide*
CB40	*IBM 2780/3780 Workstation Facility User's Guide*
CB41	*HASP Workstation Facility User's Guide*
CB42	*Level 66 Host Resident Facility User's Guide*
CB43	*Terminal Concentration Facility User's Guide*

In addition, the following documents provide general hardware information:

Order No.	*Manual Title*
AS22	*Honeywell Level 6 Minicomputer Handbook*
AT04	*Level 6 System and Peripherals Operation Manual*
AT97	*MLCP Programmer's Reference Manual*
FQ41	*Writable Control Store User's Guide*

Contents

Figures

Tables

Section 1
Introduction

The GCOS 6 Mod 400 operating system for the Level 6 minicomputers provides a comprehensive set of system services which form a base for executing user-written applications, Honeywell-supplied applications, and program development tools. It provides an online, interrupt-driven operation for multiple users and a single, low-priority batch operation typically used for program development and associated activities.

A number of different operating environments are possible, controlled in part by options exercised at system configuration, and in part by options chosen by the system operator at startup or at various times during the operating day. These environments are more fully described in Section 2, "Operating Environments."

Access to the system by users can be achieved in a variety of ways, again depending in part on system configuration options selected. These options are concerned mainly with the definition of local and/or remote terminal devices and how they are connected to the system. These are described in Section 3, "User Terminal Startup." Other access options, normally under the control of the system operator, are concerned with the procedures by which a user identifies himself (logs in) to the system through a connected terminal. This subject is treated in Section 4, "User Access to the System."

The remaining sections comprise descriptions and examples of the use of various system components: the Editor (Section 5), the Assembler and Macro Preprocessor (Section 6), the COBOL Compiler (Section 7), the FORTRAN Compiler (Section 8), and the Sort component (Section 9). Each of these sections presents terminal and/or line printer listings representing the actions performed. In these listings, heading lines may vary in detail depending on the component that initiated the listing or, in some cases, may be omitted. However, in actual use, the user will see heading lines consisting of three major fields of information, as shown below.

1. System Identification: GCOS6 MOD400-$\left\{\begin{matrix}S\\L\end{matrix}\right\}$ rrr-mm/dd/hhmm

   S — SAF
   L — LAF
   rrr — Release number of the operating system
   mm/dd/hhmm — Date/time when operating system was created (month, day, hour, and minute)

2. Component Identification: xxxxx-rrrr-mm/dd/hhmm

   xxxxx — Component name
   rrrr — Revision number of component
   mm/dd/hhmm — Date/time that specified revision of component was created (month, day, hour, and minute)

3. Time of program execution: yyyy/mm/dd hhmm:ss.t

   Date/time of program execution (year, month, day, hour, minute, second, and tenth of second)

GUIDE TO USING THE MANUAL SET

A guide to the use of the manual set is provided below. Information is tailored for specific classes of users — applications programmers, systems programmers, and operators. (As used in this guide, the applications programmer writes applications programs; the system programmer configures the system and defines the environment for each application; the operator operates the system from the operator terminal.) Included as a separate subsection is a guide for those who will use the Level 6 in a distributed processing environment.

APPLICATIONS PROGRAMMER'S MANUAL GUIDE

Figure 1-1 illustrates the suggested sequence in using the manuals. If you wish to start using the system by writing an application program, begin by using the *Programmer's Guide* manual. It illustrates: (1) various ways to gain access to the system, (2) a sample Editor session, and (3) for application languages, the procedure for performing program preparation and execution. Working with the small subset of commands used in the examples is a good approach to learning the system command set. This approach for getting started assumes that a system programmer has already configured and started up a suitable application environment. While using the system, you may wish to familiarize yourself with the system facilities described in the *System Concepts* manual.

Through examples, the *Programmer's Guide* illustrates how to use the system facilities. Other manuals provide reference material. The *Program Preparation* manual contains Editor directives (statements) to create and update an application language source unit. For each of the languages the appropriate language reference manuals contain the description of the language statements. Operating system dependencies, if any, that affect how you write the application are described in the *Programmer's Guide*. If the application uses communications, refer to the *Communications Processing* manual. Read the *Data File Organizations and Formats* manual if you require a better understanding of a language-supported file organization that is to be used in an application, or if you must calculate the size of a data file. You can use Monitor macro calls, as described in the *System Service Macro Calls* manual, in assembly language programs. Before your program can be entered for execution, it must be linked as described in the *Program Execution and Checkout* manual.

For program compilation or assembly and execution, the procedures described in the *Programmer's Guide* might be sufficient. To obtain more control over the execution of your program or utilize the system facilities more completely or efficiently, use the commands described in the *Commands* manual. If you wish to use the operator terminal, read the *Operator's Guide*. In many cases, the description of commands must be supplemented by system concepts described in the *System Concepts* manual. Rather than read all the conceptual material at one time, you may find it more meaningful to refer to it in conjunction with the appropriate reference material. The *Commands* manual also describes the utilities. An assembly language program, the Patch, Debug, and Dump utilities are described in the *Program Execution and Checkout* manual; file transmission from Level 6 to a host system is described in the *File Transmission* manual appropriate to the host system. Error messages and return status codes are listed in the *System Messages* manual.

SYSTEM PROGRAMMER'S MANUAL GUIDE

Figure 1-2 illustrates the suggested sequence for using the manuals. The *System Building* manual provides you with the configuration directives (statements) and startup procedures to configure and start up a MOD 400, a Remote Batch Facility (RBF), or a Data Entry Facility (DEF) system. You must know the conceptual material in the *System Concepts* manual in order to successfully use the configuration directives. To tailor an applications environment suitable for the intended application, use the operator commands described in the *Operator's Guide* manual. Error messages are listed in the *System Messages* manual. If you are working with an application that runs under the BES operating system, the *System Concepts* manual contains MOD 400 and BES compatibility considerations.

OPERATOR'S MANUAL GUIDE

Figure 1-3 illustrates the suggested sequence for using the manuals. Specific operator job functions must be determined by each installation; a large system might have a person assigned as an operator; a small system might have each programmer also act as an operator. The *Operator's Guide* indicates the system procedures performed through the operator terminal and describes operator commands used in system operation.

The *Programmer's Guide* contains examples using commands (described in the *Commands* manual) that are similar to operator commands. The *System Concepts* manual provides an understanding of the operating system. Note that the *Operator's Guide* describes using the

operator terminal for operator functions to enter operator commands to the system task group, or for user functions to enter commands to a user task group. To run the utilities, use the commands (described in the *Commands* manual) entered through the operator terminal functioning as a user terminal. Error messages are listed in the *System Messages* manual.

GUIDE FOR USING THE MANUALS IN A DISTRIBUTED PROCESSING ENVIRONMENT

GCOS 6 Mod 400 supports the use of Level 6 in a distributed processing environment. Using Honeywell-supplied software packages, processing capabililty can be assigned to sites remote to the host computer system. With the functional links provided by Honeywell, a Level 6 can be configured as a host processor and specialized processing (i.e., forms data entry) assigned to remote terminals. Also, the user can develop links with non-Level 6 host processors and distribute the total processing load between the host and Level 6.

The software packages available to the user include the Data Entry Facility, Remote Batch Facility, Terminal Concentration Facility, File Transmission Facility, Host Resident Facility (Level 66), and IBM workstation emulation software. Figure 1-4 indicates the documentation available for the operation and use of such software. Configuration information, if applicable, is contained in the *System Building* manual.

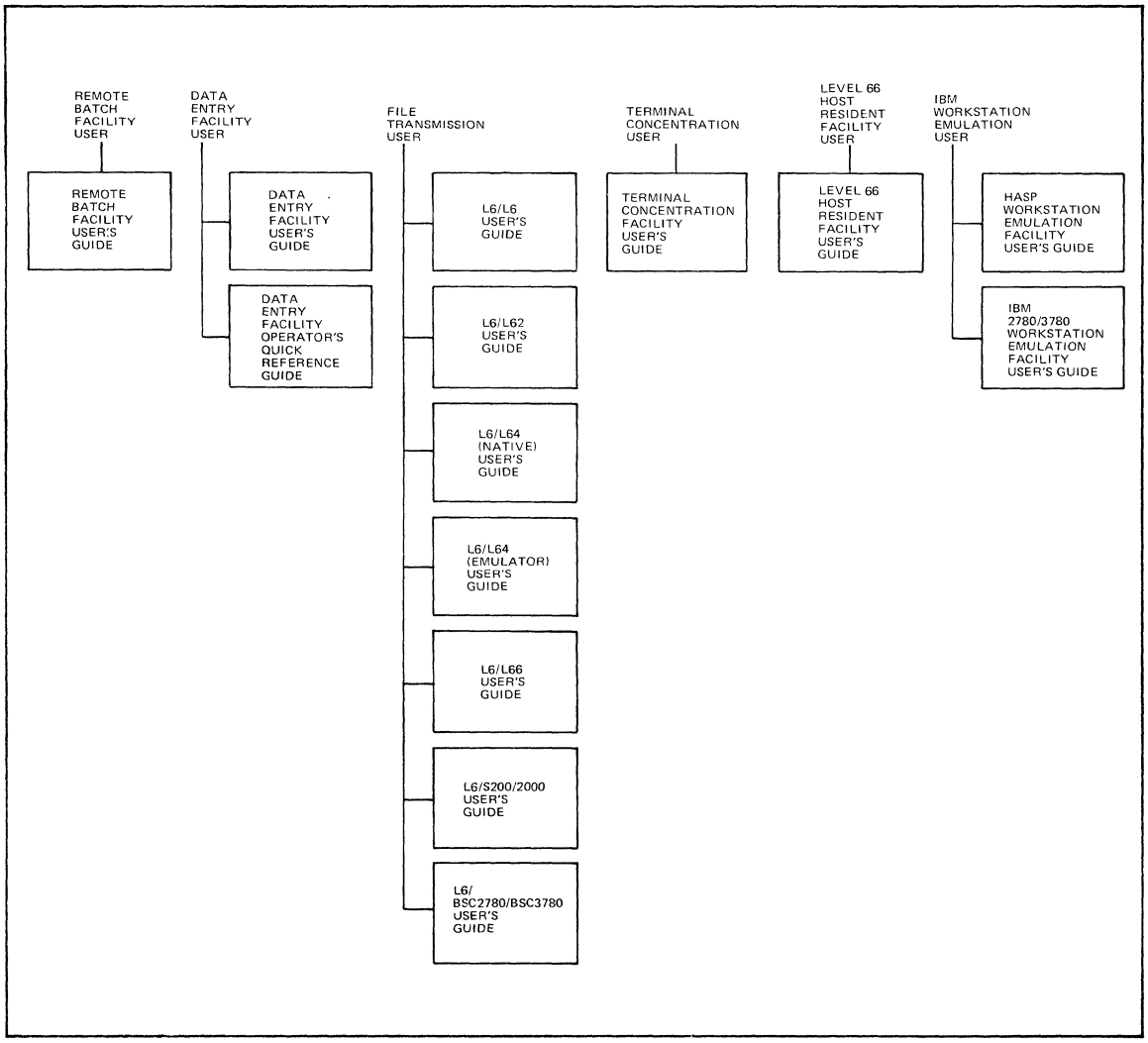

Figure 1-4. Guide for Using the Manuals in a Distributed Processing Environment

Section 2
Operating Environments

The Mod 400 operating system allows a wide variety of operating environments, ranging from a single operator-controlled configuration to one in which the operator, other users, or a combination if both can control the configuration at any time during the operating day. This range of operating environments is described in this section.

OPERATOR-ONLY ENVIRONMENT

This environment is one in which a designated operator and a limited number of users (typically programmers developing application programs) use the system on a first-come first-served basis for developing and testing programs. All work is done through the operator terminal, through either the system task group or a single online task group created by the system startup procedure. Certain functions can be performed through either of the two task groups; others can be done only through the system task group or the online task group — refer to the *Operator's Guide* and the *Commands* manuals for details on which functions can be performed from each task group.

ALL-ONLINE ENVIRONMENT

An all-online environment is one in which one or more users can concurrently use the facilities of the operating system to perform interactive tasks of any kind permitted by the command language described in the *Commands* manual, plus any user applications that can be invoked through the command processor. This latter category consists of user programs in the form of bound units that are called from a task group in which the command processor is declared as the lead task when the task group is created. A task group can also be created by the operator or another online user, declaring the application bound unit as the lead task; in this case the creation of the task group and its activation results directly in the execution of the declared bound unit, without the need to enter its name as a command.

An example of this kind of environment is one in which several task groups have the command processor as lead task and one or more other task groups have specific application programs such as the Data Entry Facility and user-created programs as lead tasks. The former task groups can be used for editing source program files, entering requests for jobs to be run in the batch task group (see below), requesting printouts of files, etc. Concurrent with these activities can be the execution of the user application programs constituting the latter set of task groups. From the user's point of view, each task group has the appearance of having control of the system.

ONLINE/BATCH ENVIRONMENT

This environment differs from the all-online environment only in that, in addition to the creation of the online task groups, a batch task group has also been created by the designated operator from the operator terminal. Once this task group has been created, any online task group having the command processor as its lead task can enter requests for jobs to be run through the use of the EBR (ENTER BATCH REQUEST) command. Typical of such batch jobs would be requests for compilations, links, application program checkout runs, and the like.

Creation and utilization of the batch task group requires the existence of at least the designated operator terminal, through which the batch task group is created and through which requests to it can be entered. Jobs run in the batch task group are normally controlled by a previously created file containing commands directing the execution of the jobs, and not by interactive dialog from a terminal. Section 4 contains additional information on the use of the batch task group.

DEDICATED APPLICATION ENVIRONMENT

This is an environment in which system startup or operator action subsequent to startup results in the creation of one or more task groups in which a user application, and not the command processor, is the lead task. In such an environment no interactive processing using system commands takes place; rather, whatever processing occurs is dependent on the nature of the application — e.g., data entry, an inventory application, etc.

MIXED ENVIRONMENT

The Mod 400 system does not restrict the user to any one of the foregoing environments at any given time. Given a large enough system, any of these can be combined with any others to provide concurrent interactive, batch, and dedicated operations on a selected terminal basis. That is, a selected set of terminals can be associated with interactive tasks, while others can be related to the dedicated application tasks.

Section 5
Using the Editor

Before studying this section, you should have a knowledge of the Editor operation. The Editor is described in the Program Preparation manual.

This section shows how the Editor is used to modify the contents of files, merge files and place macro routines in the macro library directory. The four files to be altered are SMPM01 (Example 1), SMPM02 (Example 2), SMPM03 (Example 3), and SMPM04 (Example 4). The directives that control the Editor are contained in a file ^SYSMAC>SMPCMDIFL, Figure 5-1. This file comprises 56 lines, some of which include multiple directives.

This session of the Editor accomplishes the following functions:

- SMPM01 and SMPM02 are altered and written to files SMMPL1 and SMMPL2.
- SMMPL1 and SMMPL2 are combined to form file SMPMAC.P containing macro statements and calls to be processed by the macro preprocessor.
- SMPM03 and SMPM04 are altered and written as files SMMPL3 and SMMPL4 respectively.
- SMMPL3 and SMMPL4 are altered and written as macro library routine files SAMPL1 and SAMPL2, respectively, into the MACRO>EXEC_LIB directory.

Example 1:

File SMPM01 before Editing

```
 1            TITLE     SMPMAC,'3/1/77' EDITOR/MACRO EXAMPLE
 2  * INSERT LN 2 LIBM STATEMNT BEFORE THIS LN..THEN DEL THIS LN
 3  SMPLM     MAC       P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(;
 4  P7=),P8=TWO,P9=$COMM,PA=A,PB=B,PC=T2,PD=SAMPLE,PE=PROG2
 5  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 6  LE         USE CHANGE FUNCTION TO ADD SETA VALUE FOR THIS LN
 7  L4         USE CHANGE FUNCTION TO ADD SETA VALUE FOR THIS LN
 8  L5         USE CHANGE FUNCTION TO ADD SETA VALUE FOR THIS LN
 9  L6         USE CHANGE FUNCTION TO ADD SETA VALUE FOR THIS LN
10  * ADD L7 SETA VALUE W/CHANGE FUNCTION .. THEN DELT THIS LN
11  L8         SETA      XLOC
12  L9         SETA      XVAL
13  LA         SETA      [Z'01']
14  LB         SETA      COMM
15  LZ         SETA      ', '
16  *USED EDIT READ FUNCT TO ADD "SMPM02" PORTION TO  FILE*
```

Example 2:

File SMPM02 before Editing

```
 1  *
 2  * THESE UNPROTECTED COMMENT LINES WILL BE DROPPED
 3  * WHEN MACRO PREPROCESSED.
 4  *
 5  ?G4         #L4         ?G1         (G1 INITIAL VALUE=$)
 6              #L5         ?IX(#LE,?PE)?GN?P1
 7  ?G5         #L6         ?P3?VL(35)?P4#LZ#LA
 8              #L7         ?G4
 9              #L7         ?P6?P8?GN?AL(?PC)?P7
10              #L8         ?SS(?P4,7,1)
11              #L9         ?VP(11)
12  ?G6         #L8         ?G7
13  ?GA         #L4         ?P9+?G3
14              ENDM
15  G3          SETN        1
16  G4          SETA        'ZERO'      (APOSTROPHE'S DROPPED WHEN SUBSTI.)
17  G5          SETA        'NAME'
18  G6          SETA        '$COMM'
19  G7          SETN        100
20  GA          SETA        'COM1'
21  GB          SETA        ','
22  *
23  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SMPLM" ****
24  *
25              SMPLM,                  (CALL IN-LINE MACRO ROUTINE)
26  *
27  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SAMPL1" ****
28  *
29  CALL1       SAMPL1      1,,,,,,+,150,;
30  ,,START,$C
31  *
32  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SAMPL2" ****
33  *
34  CALL2       SAMPL2      $F,,,,,,,,;
35  ,,,,,,,,,,,,,,,,,,,,,,,,,LINK
36              END         SMPMAC,START
```

Example 3:

File SMPM03 before Editing

```
 1  SAMPL1     MAC        P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(,P7=);
 2  P8=TWO,P9=$COMM,PA=A,PB=B,PD=SAMPLE,PE=PROGRAM
 3  *
 4  *
 5  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 6  *
 7  *
 8  L4         SETB       ORG
 9  L5         SETB       DC
10  L6         SETB       LDR
11  L7         SETB       STR
12  L8         SETB       CALL
13  L9         SETB       LB
14  LA         SETB       BBT
15  LB         SETB       SLD
16  LC         SETB       '='
17  LD         SETB       (Z'32')
18  LE         SETB       'PROG2.START2(,)NAME'
19  *
20  *
21  * SET GLOBAL VALUES WITHIN MACRO ROUTINE *
22  *
23  *
24  GH         SETA       'ORG INTO COMMON'
25  GG         SETA       'ORG INTO INTERNAL LOC'
26  GC         SETA       'EXTERN VAL REFERENCE'
27  GD         SETA       'COMMON REFERENCE'
28  GE         SETA       'EXTERNAL LOCATION REFERENCE'
29  GF         SETA       'FORWARDS TEMP LABEL REFERENCE'
30  *
31  * UNPROTECTED LINES OMITTED WHEN PRE-PROCESSED
32  *
33             ?L4        ?P9                    ?GH
34             ?L5        ?VR(?P3,?PD)?GB?SR(?P4,?PE)
35             ?L4        ?G4?P7?P8              ?GG
36  ?PC        ?L6        $R1,?LC?P8                        ?GC
37             ?L7        $R1,<?GA               ?GD
38  (*)
39  ?PD        ?L6        $R1,<?PA               ?GE
40  (*)
41             ?L8        PROG2.?SS(?LE,7,6)?GBNAME
42             ?L9        ?G4?P7?P1?GB?LC?VL(13)
43             ?LA        >?P7$F                 ?GF
44             ?LB        $S1?GB?LCZ'?CH(1,-2)?CH(2,-2)?CH(3,-2)?CH(4,-2)'
45  ENDCL1     ENDM
```

Example 4:

File SMPM04 before Editing

```
 1  SAMPL2    MAC       P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(,P7=);
 2  P8=TWO,P9=$COMM,PA=A,PB=B
 3  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 4  L4        SETA      >=(Z'1300')
 5  LA        SETA      IOLD
 6  LD        SETA      $R1
 7  LE        SETA      'PROG2.START[,]NAME'
 8  LG        SETA      $OR
 9  LC        SETN      -32768
10  DELT$     DC  'DELETE LINE ENDING IN $'$
11  LP        SETN      32767
12  LN        SETN      0
13  LI        SETA      REZ
14  LY        SETA      HLT
15  LZ        SETA      ','
16  * SET GLOBAL VALUES WITHIN MACRO ROUTINE *
17  G7        SETN      -32766
18  G2        SETA      'BACKWARDS TEMP LABEL REFERENCE'
19  G5        SETA      CTRL
20  *
21  * UNPROTECTED LINES OMITTED WHEN PRE-PROCESSED
22  *
23  ?P1       ?LA       ?P5?LZ?L4,=?LD
24  ?P1       ?LA       ?P5?LZ?L4,=?LD
25            ?LG       ?LD,?VG(3)
26            ?LI       ?LD,-$C    ?G2
27            ?LY
28            ?G5       ?PZ ?SS(?LE,1,5)
29  ^DEL      DC  'DELETE LINE BEGINNING IN ^'
30            IFNE      ?G7,?LP,GTEND
31            FAIL
32  GTEND     GOTO      ENDIT
33  DLET      DC  'DELETE LINE BEFORE QUIT'
34  ENDCL2    ENDM
```

```
 1  R ^SYSMAC>SMPM01
 2  X
 3  6,9CLE     SETA       ' PROG2.START2[,]NAME'
 4  L4         SETA       EQU
 5  L5         SETA       RESV
 6  L6         SETA       TEXT
 7  L7         SETA       XDEF
 8  !F
 9  2I         LIBM       'EXEC_LIB',SAMPL1,SAMPL2!F!?
10  =.-1;$K(SMMPL1)
11  X
12  1,$DX
13  R SMPM02
14  1,13V!P/#L/.-12;13G!P/#L/.-9;13S'#L'?L'P=
15  1,$M(SMMPL2)X
16  R SMPM03
17  8,17S/SETB/SETA/8,17P
18  1,$M(SMMPL3)X
19  R ^SYSMAC>SMPM04
20  X29A       IFE        ?G7,?LC,IFE1
21             FAIL
22  ENDIT      IFNL       ?P2,?LC,*
23  IFE1       NULL!F
24  /SS..LE/!P/$$/!PD/^^/!PD
25  1,$K(SMMPL4)
26  X1,$D
27  B(SMMPL1)
28  W ^SYSMAC>SMMPL1.IN.A
29  1,$DX
30  B(SMMPL2)
31  W ^SYSMAC>SMMPL2.IN.A
32  1,$D
33  B(SMMPL3)
34  W ^SYSMAC>SMMPL3
35  1,$D
36  B(SMMPL4)
37  W ^SYSMAC>SMMPL4
38  1,$DX
39  R SMMPL1.IN.A
40  /INSERT/!PD/ADD L7/!PD
41  15R SMMPL2.IN.A
42  X52!PD
43  E F0 >SPD>LPT00
44  1,$!PW SMPMAC.P
45  1,$D
46  R SMMPL3
47  X/L4/;/LE/S/SETB/SETA/
48  1,$!PW ^SYSRES>LDD>MACRO>EXEC_LIB>SAMPL1
49  1,$D
50  R SMMPL4
51  /DLET/D
52  Q
53  1,$!PW ^SYSRES>LDD>MACRO>EXEC_LIB>SAMPL2
54  X
55  E F0
56  Q
```

Figure 5-1. Sample Editor Directives in File SMPCMDIFL

OPERATOR TERMINAL TYPEOUT

The typeout produced at the operator's terminal during the editing process is shown in Figure 5-2. In this figure, the editor directive line that produced each line of the typeout is indicated by the circled numbers at the left of the typeout lines. Note that not every directive generates a typeout; e.g., the R (read) and W (write) directives. The lines of typeout that are produced in the editing of each of the four input files are indicated by brackets in the left margin.

In the typeout, the response to the Editor directives begins after the line ($H)EDIT-0100-11/21/0827.

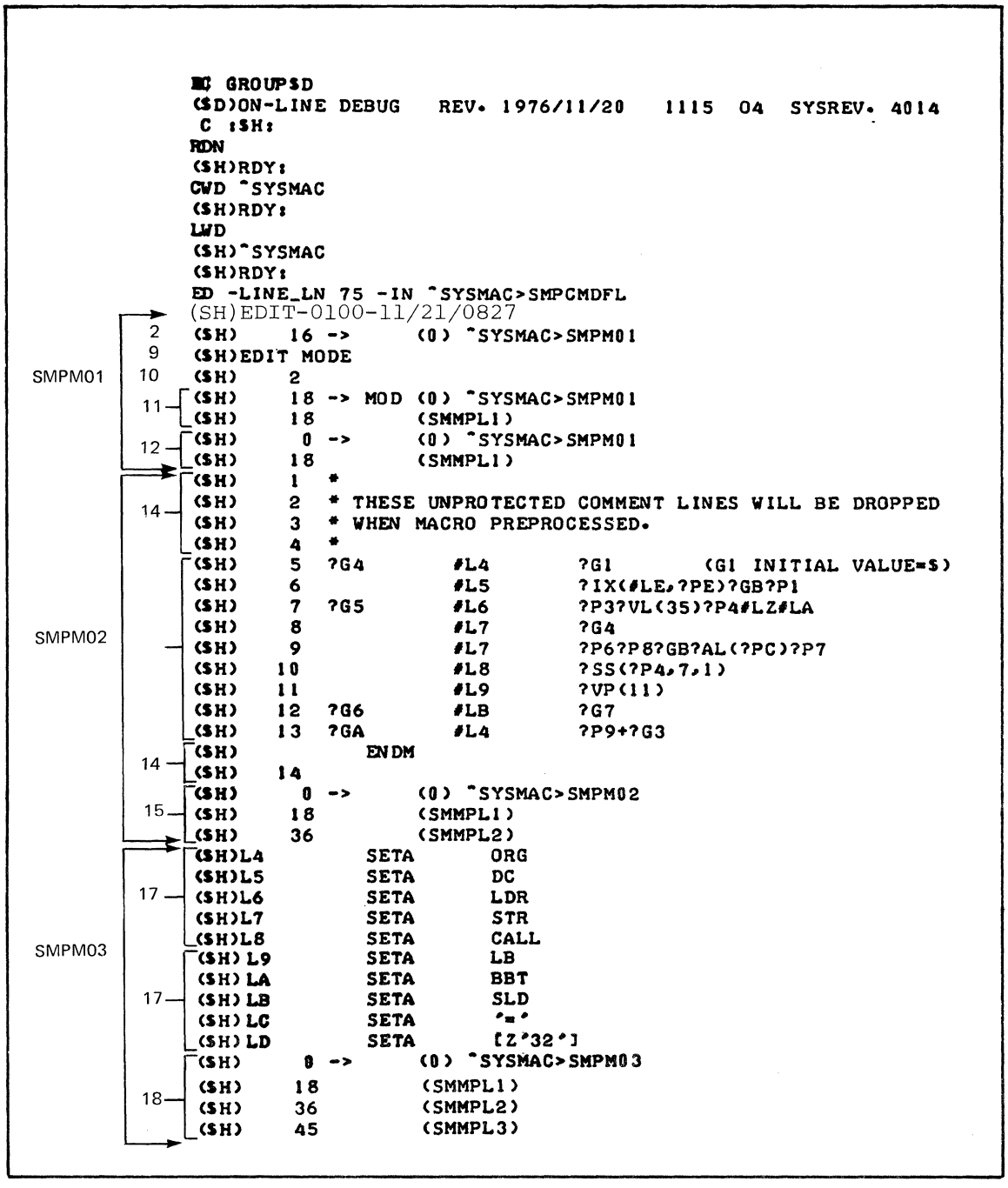

Figure 5-2. Terminal Responses from Sample Editor Directives of Figure 5-1

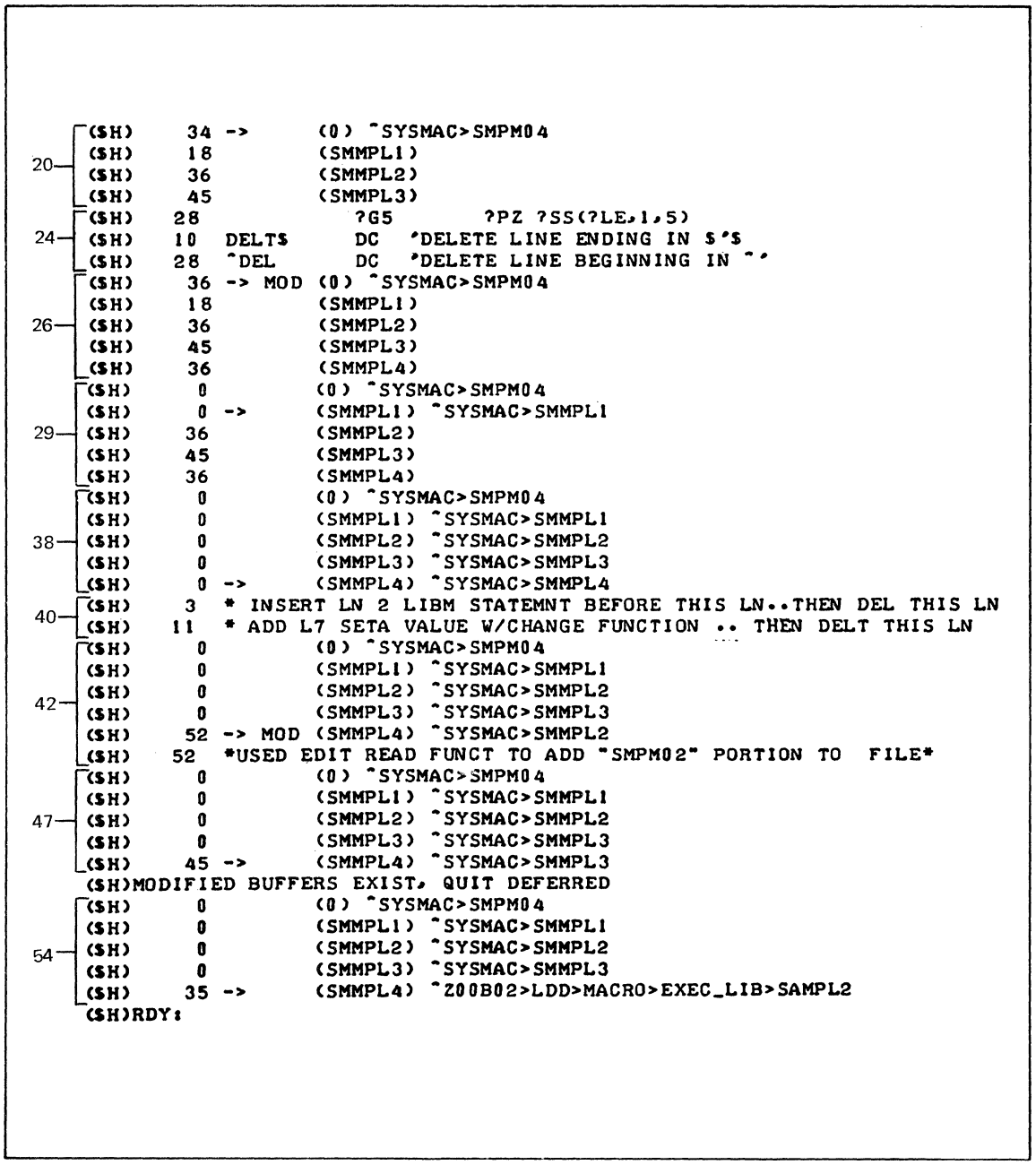

Figure 5-2 (cont). Terminal Responses from Sample Editor Directives of Figure 5-1

EXPLANATION OF EDITOR ACTIONS

Table 5-1 describes the actions performed by the Editor as it processes the directive file. The entries under the heading Line No. designate lines in the directive file Figure 5-1. The entries under the heading Terminal Typeout are abbreviated versions of those in Figure 5-2. For lines with multiple directives, they show which directive caused the typeout.

The assembly language program with unexpanded macro calls created from Example 1 and Example 2 is shown in Figure 5-3. The directive that caused it to be printed is contained in line 44.

Figures 5-4 and 5-5 are the unexpanded macro routines created from Example 3 and Example 4 respectively. The directives that caused them to be printed are included in lines 48 and 53 respectively.

TABLE 5-1. EXPLANATION OF EDITOR ACTIONS

Line No.	Editor Directive Description	Terminal Typeout
1	R ^SYSMAC>SMPM01 Read the 16-line file (example 1) into the current buffer (0).	
2	X Display the status of the current buffer (denoted by →). Sixteen lines were read into current buffer (0).	16->(0) ...
3	6,9Ctext Change lines 6 through 9 of the buffer with this text for line 6.	
4	Text for line 7.	
5	Text for line 8.	
6	Text for line 9.	
7	Text, this additional line is inserted after the previous four lines were changed. Note that the Editor recognizes tab characters.	
8	!F Terminate input mode and enter edit mode.	
9	2Itext!F!? Insert text before line 2. Terminate input mode. Display current mode.	 EDIT MODE
10	=.-1;$K(SMMPL1) Display current line pointer. Move the current line pointer back one line to a new current line pointer position. Copy the lines from that position to the last line in the current buffer into auxiliary buffer, SMMPL1.	 2
11	X Display status of current and auxiliary buffers. There are 18 lines in current buffer (0), which has been modified (MOD) since it was read in, and 18 lines in auxiliary buffer SMMPL1.	 18-> MOD (0) ... 18 (SMMPL1)
12	1,$DX Delete the first through last line of the current buffer (0). Display status of buffers.	 0->(0) ... 18 (SMMPL1)
13	R SMPM02 Read 36-line file (example 2) into the current buffer (0).	
14	1,13V!P/#L/.-12;13G!P/#L/.-9;.S'#L'?L'!P= For lines 1 through 13, display line numbers and all lines that do not contain the expression #L. Move the current line pointer back 12 lines from line 14 to line 2. For lines 2 through 13, display all lines and their line numbers containing the expression #L.	 1* 2*THESE ... . . . 4* 5?G4#L4 ... . . . 13?GA #L4 ...

TABLE 5-1 (CONT). EXPLANATION OF EDITOR ACTIONS

Line No.	Editor Directive Description	Terminal Typeout
	Move the current line pointer back nine lines to line 5 and substitute ?L for #L, in that line and the next 8 lines. Note that the apostrophe is used as a delimiter.	
	Print current line.	ENDM
	Print current line pointer value.	14
15	1,$M(SMMPL2)X	
	Move line 1 through the last line of the current buffer to auxiliary buffer SMMPL2. The contents of the current buffer (0) are erased. Display the status of the buffers.	0->(0) ... 18 (SMMPL1) 36(SMMPL2)
16	R SMPM03	
	Read 45-line file (example 3) into current buffer (0).	
17	8,17S/SETB/SETA/8,17P	
	For lines 8 through 17, substitute SETA for SETB.	
	Print lines 8 through 17 without line numbers.	L4 SETA ... . . . LD SETA ...
18	1,$M(SMMPL3)X	
	Move line 1 through last line of the current buffer into auxiliary buffer SMMPL3 and erase buffer (0).	
	Display buffer status.	0->(0) ... 18 (SMMPL1) 36 (SMMPL2) 45 (SMMPL3)
19	R ^SYSMAC>SMPM04	
	Read the 34-line file (example 4) into the current buffer (0).	
20	X29A	
	Display buffer status, 34 lines are currently in buffer (0).	34->(0) ... 18 (SMMPL1) 36 (SMMPL2) 45 (SMMPL3)
	After line 29, append four lines of text. Text for line 30.	
21	Text for line 31.	
22	Text for line 32.	
23	text!F	
	Last line of text (line 33).	
	Terminate input mode and enter edit mode.	
24	/SS..LE/!P/$$/!PD/^ ^/!PD	
	Search the current buffer for the first occurrence of the expression SS..LE, where .. are any two characters.	
	List the line and its line number.	28?G5 ?PZ ?SS(?LE...

TABLE 5-1 (CONT). EXPLANATION OF EDITOR ACTIONS

Line No.	Editor Directive Description	Terminal Typeout
	Locate a line that ends with $ as the last character. List the line and its number; then delete the line.	10DELT$ DC...'D...$'$
	Locate a line beginning with ^. List, then delete the line and its line number.	28 ^ DEL DC 'DELETE ...
25	1,$K(SMMPL4) Copy the current buffer contents from first through last line into auxiliary buffer SMMPL4.	
26	X1,$D Display the status of the buffers.	36->MOD(0) ... 18 (SMMPL1) 36 (SMMPL2) 45 (SMMPL3) 36 (SMMPL4)
	Delete first through last line of current buffer.	
27	B(SMMPL1) The auxiliary buffer, SMMPL1, is made the current buffer prior to writing.	
28	W ^SYSMAC>SMMPL1 Write the current buffer contents as a file whose pathname is ^SYSMAC>SMMPL1.	
29	1,$DX Delete the first through last line of the current buffer. Display the buffer status. The pointer points to current buffer, SMMPL1.	0(0) ... 0->(SMMPL1) ... 36 (SMMPL2) 45 (SMMPL3) 36(SMMPL4)
30	B(SMMPL2) The auxiliary buffer, SMMPL2, is made the current buffer prior to writing.	
31	W ^SYSMAC>SMMPL2 Write the current buffer contents as a file whose pathname is ^SYSMAC>SMMPL2.	
32	1,$D Delete the first through last line of the current buffer.	
33	B(SMMPL3) The auxiliary buffer, SMMPL3, is made the current buffer prior to writing.	

TABLE 5-1 (CONT). EXPLANATION OF EDITOR ACTIONS

Line No.	Editor Directive Description	Terminal Typeout
34	W ^SYSMAC>SMMPL3 Write the current buffer contents as a file whose pathname is ^SYSMAC>SMMPL3.	
35	1,$D Delete the first through last line of the current buffer.	
36	B(SMMPL4) The auxiliary buffer, SMMPL4, is made the current buffer prior to writing.	
37	W ^SYSMAC>SMMPL4. Write the current buffer contents as a file whose pathname is ^SYSMAC>SMMPL4.	
38	1,$DX Delete the first through last line of the current buffer. Display the status of the buffers. SMMPL4 is the current buffer. All the buffers have been cleared.	 0 (0) ... 0 (SMMPL1) .. 0 (SMMPL2) .. 0 (SMMPL3) .. 0->(SMMPL4) ...
39	R SMMPL1 Read the file SMMPL1 into the current buffer, SMMPL4.	
40	/INSERT/!PD/ADD L7/!PD Locate the first line containing the expression, INSERT, list it and its line number, and then delete it. Starting at the current line, locate the first line containing the expression, ADD L7, list it and its line number, and then delete it.	 3* INSERT LN ... 11* ADD L7 SETA ...
41	15R SMMPL2 Read the file, SMMPL2, into the current buffer after line 15 of the buffer. Two files are being merged.	
42	X52!PD Display the status of the buffers. Current buffer, SMMPL4, now has 52 lines. List line 52 then delete it.	 0 (0) ... . . . 52->MOD(SMMPL4)... 52 *USED EDIT ...
43	E FO >SPD>LPT00 The Execute directive allows you to execute the ECL command FO to change the output file from the operator's terminal to the line printer.	
44	1,$!PW SMPMAC.P List the first through last line of current buffer on the line printer (Figure 5-3). Write the current buffer as a file whose pathname is SMPMAC.P.	

TABLE 5-1 (CONT). EXPLANATION OF EDITOR ACTIONS

Line No.	Editor Directive Description	Terminal Typeout
45	1,$D Delete the first through last line of the current buffer.	
46	R SMMPL3 Read the file, SMMPL3, into the current buffer, SMMPL4.	
47	X/L4/;/LE/S/SETB/SETA/ Display the status of the buffers.	0 (0) ... 0 (SMMPL1) ... 0 (SMMPL2) ... 0 (SMMPL3) ... 45->(SMMPL4) ...
	Locate the first line containing the expression, L4. Starting with the line containing L4 through the line containing the expression LE, substitute SETA for all occurrences of SETB.	
48	1,$!PW ^Z00B02>LDD>MACRO>EXEC_LIB>SAMPL1 List the first through last line of the current buffer on the line printer (Figure 5-4).	
	Write the current buffer as a library routine file whose pathname is ^Z00B02>LDD>MACRO>EXEC_LIB>SAMPL1.	
49	1,$D Delete the first through last line of the current buffer.	
50	R SMMPL4 Read the file, SMMPL4, into the current buffer.	
51	/DLET/D Locate and delete the line containing the expression DLET.	
52	Q Quit. The quit is deferred since a buffer has been modified and has not been written to a file. You have *one* more chance to write the contents of the current buffer as a file.	MODIFIED BUFFERS EXIST ...
53	1,$!PW ^Z00B02>LDD>MACRO>EXEC_LIB>SAMPL2 List the first through last line of the current buffer on the line printer (Figure 5-5). Write the current buffer contents as a library routine file whose pathname is ^Z00B02>LDD>MACRO>EXEC_LIB>SAMPL2.	
54	X Display buffer status. Status is always displayed on the operator's terminal even though the output file is the printer.	0 (0) ... 0 (SMMPL1) ... 0 (SMMPL2) ... 0 (SMMPL3) ... 35->(SMMPL4)
55	E FO The Execute directive allows you to execute the ECL command FO to change the output file from the line printer back to the operator's terminal.	
56	Q Quit. Exit from the Editor.	

```
 1              TITLE     SMPMAC,'3/1/77' EDITOR/MACRO EXAMPLE
 2              LIBM      'EXEC~LIB',SAMPL1,SAMPL2
 3  SMPLM       MAC       P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(;
 4  P7=),P8=TWO,P9=$COMM,PA=A,PB=B,PC=T2,PD=SAMPLE,PE=PROG2
 5  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 6  LE          SETA      ' PROG2.START2[,]NAME'
 7  L4          SETA      EQU
 8  L5          SETA      RESV
 9  L6          SETA      TEXT
10  L7          SETA      XDEF
11  L8          SETA      XLOC
12  L9          SETA      XVAL
13  LA          SETA      [Z'01']
14  LB          SETA      COMM
15  LZ          SETA      ', '
16  *
17  * THESE UNPROTECTED COMMENT LINES WILL BE DROPPED
18  * WHEN MACRO PREPROCESSED.
19  *
20  ?G4         ?L4       ?G1       (G1 INITIAL VALUE=$)
21              ?L5       ?IX(?LE,?PE)?GB?P1
22  ?G5         ?L6       ?P3?VL(35)?P4?LZ?LA
23              ?L7       ?G4
24              ?L7       ?P6?P8?GB?AL(?PC)?P7
25              ?L8       ?SS(?P4,7,1)
26              ?L9       ?VP(11)
27  ?G6         ?LB       ?G7
28  ?GA         ?L4       ?P9+?G3
29              ENDM
30  G3          SETN      1
31  G4          SETA      'ZERO'    (APOSTROPHE'S DROPPED WHEN SUBSTI.)
32  G5          SETA      'NAME'
33  G6          SETA      '$COMM'
34  G7          SETN      100
35  GA          SETA      'COM1'
36  GB          SETA      ','
37  *
38  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SMPLM" ****
39  *
40              SMPLM,                (CALL IN-LINE MACRO ROUTINE)
41  *
42  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SAMPL1" ****
43  *
44  CALL1       SAMPL1    1,,,,,,+,150,;
45  ,,START,$C
46  *
47  **** THE FOLLOWING PORTION OF CODE IS ADDED FROM "SAMPL2" ****
48  *
49  CALL2       SAMPL2    $F,,,,,,,,;
50  ,,,,,,,,,,,,,,,,,,,,,,,,,LINK
51              END       SMPMAC,START
```

Figure 5-3. Sample of Unexpanded Assembly Language Program with Macro Calls and Statements (SMPMAC.P)

```
 1  SAMPL1    MAC       P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(,P7=);
 2  P8=TWO,P9=$COMM,PA=A,PB=B,PD=SAMPLE,PE=PROGRAM
 3  *
 4  *
 5  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 6  *
 7  *
 8  L4         SETA      ORG
 9  L5         SETA      DC
10  L6         SETA      LDR
11  L7         SETA      STR
12  L8         SETA      CALL
13  L9         SETA      LB
14  LA         SETA      BBT
15  LB         SETA      SLD
16  LC         SETA      '='
17  LD         SETA      (Z'32']
18  LE         SETA      'PROG2.START2[,]NAME'
19  *
20  *
21  * SET GLOBAL VALUES WITHIN MACRO ROUTINE *
22  *
23  *
24  GH         SETA      'ORG INTO COMMON'
25  GG         SETA      'ORG INTO INTERNAL LOC'
26  GC         SETA      'EXTERN VAL REFERENCE'
27  GD         SETA      'COMMON REFERENCE'
28  GE         SETA      'EXTERNAL LOCATION REFERENCE'
29  GF         SETA      'FORWARDS TEMP LABEL REFERENCE'
30  *
31  * UNPROTECTED LINES OMITTED WHEN PRE-PROCESSED
32  *
33             ?L4       ?P9                 ?GH
34             ?L5       ?VR(?P3,?PD)?GB?SR(?P4,?PE)
35             ?L4       ?G4?P7?P8           ?GG
36  ?PC        ?L6       $R1,?LC?PB                    ?GC
37             ?L7       $R1,<?GA            ?GD
38  [*]
39  ?PD        ?L6       $R1,<?PA            ?GE
40  [*]
41             ?L8       PROG2.?SS(?LE,7,6)?GBNAME
42             ?L9       ?G4?P7?P1?GB?LC?VL(13)
43             ?LA       >?P7$F              ?GF
44             ?LB       $S1?GB?LCZ'?CH(1,-2)?CH(2,-2)?CH(3,-2)?CH(4,-2)'
45  ENDCL1     ENDM
```

Figure 5-4. Sample of Unexpanded Macro Routine (SAMPL1) Contained in EXEC_LIB Directory

```
 1  SAMPL2     MAC        P1=0,P2=2,P3='SAMPLE',P4='PROGRAM',P5=ZERO,P6=(,P7=);
 2  P8=TWO,P9=$COMM,PA=A,PB=B
 3  * SET LOCAL VALUES WITHIN MACRO ROUTINE *
 4  L4         SETA       >=[Z'1300']
 5  LA         SETA       IOLD
 6  LD         SETA       $R1
 7  LE         SETA       'PROG2.START[,]NAME'
 8  LG         SETA       SOR
 9  LC         SETN       -32768
10  LP         SETN       32767
11  LN         SETN       0
12  LI         SETA       BEZ
13  LY         SETA       HLT
14  LZ         SETA       ','
15  * SET GLOBAL VALUES WITHIN MACRO ROUTINE *
16  G7         SETN       -32768
17  G2         SETA       'BACKWARDS TEMP LABEL REFERENCE'
18  G5         SETA       CTRL
19  *
20  * UNPROTECTED LINES OMITTED WHEN PRE-PROCESSED
21  *
22  ?P1        ?LA        ?P5?LZ?L4,=?LD
23  ?P1        ?LA        ?P5?LZ?L4,=?LD
24             ?LG        ?LD,?VG(3)
25             ?LI        ?LD,-$C    ?G2
26             ?LY
27             ?G5        ?PZ ?SS(?LE,1,5)
28             IFE        ?G7,?LC,IFE1
29             FAIL
30  ENDIT      IFNL       ?P2,?LC,*
31  IFE1       NULL
32             IFNE       ?G7,?LP,GTEND
33             FAIL
34  GTEND      GOTO       ENDIT
35  ENDCL2     ENDM
```

Figure 5-5. Sample of Unexpanded Macro Routine (SAMPL2) Contained in EXEC_LIB Directory

EXECUTING

The internal file names 0A and 0C translate to logical file numbers 01 and 03, respectively, and must be associated with the pathnames or the physical devices through a GET or ASSOC command. To execute the program, enter CARDIN.

Enter the following commands:

```
GET 01  >SPD>CDR00
GET 03  ^VOL03>FILES>OLD_MASTER
CARDIN
```

SAMPLE COBOL TERMINAL SESSION (AC8111)

Figure 7-1 illustrates an operator terminal session in which a system is configured, and the COBOL program AC8111 is compiled, linked, and executed on that system. The Entry Level COBOL compiler is specified in this session. To secify the Intermediate COBOL compiler, change the COBOL command and the LINKER LIB directive as follows.

```
COBOLI AC8111 -LO -COUT >SPD>LPT00
LIB  ^ZSYS51>ZCIRT;LINK AC8111;MAP;QT
```

The LINKER LIB directive directs the Linker to search the secondary directory for COBOL run-time routines required for linking. To execute the program, enter AC8111.

```
COBOLI ^STCOB1>SOURCE>ACC208>AC8111 -LO -XREF -COUT >SPD>LPT00
($H) COBOLI  0200  05/08/1050
($H) 0000 ERRORS
($H)END COMPILATION
($H)RDY:
LINKER AC8111 -COUT >SPD>LPT00
($H)LINKER-0100-04/05/0704
LIB <ZCIRT
LINK Z@AC8111;MP;QT
($H)ROOT  AC8111
($H)LINK DONE
($H)RDY:
AC8111
($H)Q208NUA011001
($H)Q208NUA011001          1 2 3 4
($H)                       P P P P
($H)RDY:
```

Figure 7-1. Sample Terminal Session (AC8111)

Figure 7-2 is a listing of the program AC8111, its compiled object text, and the output from the Linker. The program was compiled using the entry-level compiler.

```
=GCOS6 MOD400-S110-04/26/0916  COBOL I  0200  -LO  XREF  AC8111  78/05/08
SOURCE  PROGRAM

    1          IDENTIFICATION DIVISION.
    2         *PROGRAM Q208A01101.COBOL FROM Q208ACC.ARCHIVE.
    3          PROGRAM-ID. AC8111.
    4          ENVIRONMENT DIVISION.
    5          CONFIGURATION SECTION.
    6          SOURCE-COMPUTER. LEVEL-6.
    7          OBJECT-COMPUTER. LEVEL-6 PROGRAM COLLATING SEQUENCE IS ASCII.
    8          DATA DIVISION.
    9          WORKING-STORAGE SECTION.
   10          01  QDSPLYREC.
   11              05  QDSPLYFIX.
   12                  10  FILLER     PIC X(13)  VALUE "Q208NUA011001".
   13                  10  QTCASE     PIC XX     VALUE SPACES.
   14                  10  FILLER     PIC XX     VALUE SPACES.
   15                  10  QSTATUS    PIC XX     VALUE SPACES.
   16                  10  FILLER     PIC XX     VALUE SPACES.
   17              05  QDSPLYVBL.
   18                  10  QACTRESLT PIC X(12)  VALUE SPACES.
   19                  10  FILLER     PIC XX     VALUE SPACES.
   20                  10  QEXPRESLT PIC X(12)  VALUE SPACES.
   21                  10  FILLER     PIC XX     VALUE SPACES.
   22          01  SUMMARYS.
   23              05  SUM-LINE      PIC X(7)   VALUE "1 2 3 4".
   24              05  RESULTS.
   25                  10  TEST1R    PIC XX.
   26                  10  TEST2R    PIC XX.
   27                  10  TEST3R    PIC XX.
   28                  10  TEST4R    PIC XX.
   29         * * * TEST GO TO--FORWARD AND BACK * * *
   30          PROCEDURE DIVISION.
   31          ANFANG.
   32              DISPLAY QDSPLYFIX.
   33              GO      TO  PARA-3.
   34          WBA1.
   35              MOVE   "GO TO PARA-3"   TO  QEXPRESLT.
   36              MOVE   "FELL THRU"      TO  QACTRESLT.
   37              MOVE   "01"  TO    QTCASE.
   38              MOVE   "F"   TO    TEST1R.
   39              DISPLAY QDSPLYREC.
   40          PARA-1.
   41              MOVE   "P"   TO    TEST3R.
   42              GO      TO   EOJ1.
   43          WBA2.
   44              MOVE   "GO TO EOJ1"   TO  QEXPRESLT.
   45              MOVE   "FELL THRU"    TO  QACTRESLT.
   46              MOVE   "04"  TO  QTCASE.
   47              MOVE   "F"   TO  TEST4R.
   48              DISPLAY QDSPLYREC.
   49          PARA-2.
   50              MOVE   "P"   TO  TEST2R.
   51              GO      TO   PARA-1.
   52          WBA3.
   53              MOVE   "GO TO PARA-1" TO  QEXPRESLT.
   54              MOVE   "FELL THRU"    TO  QACTRESLT.
   55              MOVE   "03"  TO  QTCASE.
   56              MOVE   "F"   TO  TEST3R.
   57              DISPLAY QDSPLYREC.
   58          PARA-3.
   59              MOVE   "P"   TO  TEST1R.
   60              GO     TO    PARA-2.
   61          WBA4.
   62              MOVE  "GO TO PARA-2"  TO  QEXPRESLT.
   63              MOVE  "FELL THRU"     TO  QACTRESLT.
```

Figure 7-2. Sample Listings for AC8111

```
64              MOVE  "02"  TO  QTCASE.
65              MOVE  "F"   TO  TEST2R.
66              DISPLAY QDSPLYREC.
67          EOJ1.
68              MOVE   "P"  TO  TEST4R.
69              MOVE    SPACES  TO   QTCASE.
70              MOVE    SPACES  TO   QSTATUS.
71              DISPLAY QDSPLYFIX SUM-LINE.
72              MOVE    SPACES  TO    QDSPLYFIX.
73              DISPLAY QDSPLYFIX    RESULTS.
74              STOP RUN.
75          END COBOL.

 =GCOS6 MOD400-S110-04/26/0916  COBOL1   0200   -LO   XREF   AC8111   78/05/08
DATA ALLOCATION MAP
 LEVEL NO.   NAME                 LHAD  AL  PICTURE

WORKING-STORAGE SECTION
      01     QDSPLYREC            0000      X(000049)
          05 QDSPLYFIX            0000      X(000021)
          10 FILLER               0000      X(000013)
          10 QTCASE               0006   H  X(000002)
          10 FILLER               0007   H  X(000002)
          10 QSTATUS              0008   H  X(000002)
          10 FILLER               0009   H  X(000002)
          05 QDSPLYVBL            000A   H  X(000028)
          10 QACTRESLT            000A   H  X(000012)
          10 FILLER               0010   H  X(000002)
          10 QEXPRESLT            0011   H  X(000012)
          10 FILLER               0017   H  X(000002)
      01     SUMMARYS             0019      X(000015)
          05 SUM-LINE             0019      X(000007)
          05 RESULTS              001C   H  X(000008)
          10 TEST1R               001C   H  X(000002)
          10 TEST2R               001D   H  X(000002)
          10 TEST3R               001E   H  X(000002)
          10 TEST4R               001F   H  X(000002)
NO DIAGNOSTICS
```

(DATA ALLOCATION MAP)

(PICTURE)

(HALF-WORD INDICATOR; DESIGNATED BY H)

(STARTING ADDRESS OF DATA)

(DATA NAME)

(GROUP AND ELEMENTARY ITEM LEVEL NUMBERS)

(GROUP LEVEL NUMBERS)

Figure 7-2 (cont). Sample Listings for AC8111

```
=GCOS6 MOD400-S110-04/26/0916 COBOLI  0200 -LO  XREF  AC8111  78/05/08
OBJECT  CODE

STATEMENT NUMBER   30
0012  D380 0000   LNJ  $B5,<ZCRTER
0015  D380 0000   LNJ  $B5,<ZCSTOP
0018  8AC6 0000   INC
001A  0F83        B
001B  8746 0000   CL
001D  DFC6 0000   STB
001F  FFC6 0000   STB
0021  FBC6 0000   LAB
0023  8747 009A   CL
STATEMENT NUMBER   31
0000  0F81 0024   B
STATEMENT NUMBER   32
0025  9BC7 FFDA   LAB
0027  9870 0000   LDR
0029  E870 0015   LDR
002B  D380 0000   LNJ  $B5,<ZCRTY1
STATEMENT NUMBER   33
002F  4381 FFDD   CSNB
STATEMENT NUMBER   34
0002  0F81 002D   B
STATEMENT NUMBER   35
0030  0F87        B
0031  474F        DC
0032  2054        DC
0033  4F20        DC
0034  5041        DC
0035  5241        DC
0036  2D33        DC
0037  0021        ALR
0038  4C08 FFF8   DD1
003A  CC07 FFEB   DD2
```

(PARTIAL OBJECT LISTING)

(SUBROUTINE CALL)

(INSTRUCTION MNEMONIC)

(INSTRUCTION)

(LOCATION OF INSTRUCTION)

```
=GCOS6 MOD400-S110-04/26/0916 COBOLI  0200 -LO  XREF  AC8111  78/05/08
CROSS REFERENCE LISTING
NUMBER  DATA NAME                        REFERENCE NUMBERS

   31  ANFANG            NO REFERENCES
   67  EOJ1                  42
   40  PARA-1                51
   49  PARA-2                60
   58  PARA-3                33
   18  QACTRESLT             36      45      54      63
   11  QDSPLYFIX             32      71      72      73
   10  QDSPLYREC             39      48      57      66
   17  QDSPLYVBL         NO REFERENCES
   20  QEXPRESLT             35      44      53      62
   15  QSTATUS               70
   13  QTCASE                37      46      55      64      69
   24  RESULTS               73
   23  SUM-LINE              71
   22  SUMMARYS          NO REFERENCES
   25  TEST1R                38      59
   26  TEST2R                50      65
   27  TEST3R                41      56
   28  TEST4R                47      68
```

Figure 7-2 (cont). Sample Listings for AC8111

```
   34  WBA1                NO REFERENCES
   43  WBA2                NO REFERENCES
   52  WBA3                NO REFERENCES
   61  WBA4                NO REFERENCES


LINKER-0100-11/23/1258                    GCOS6 MOD400-S100-11/29/0620
RU= AC8111       LINKED ON: 1901/01/01 0002:44.1  -SAF

AC8111  01/01/01
 COBOL REV. 0200 DATE 01/01/01 TIME 0000 .

ZCRTYU  770208
HRS ASSEMBLER 2.49  06/02/77  1340.3 EDT THU
(C)  COPYRIGHT 1976 BY HONEYWELL INFORMATION SYSTEMS INC

ZCSTOP  770208
HRS ASSEMBLER 2.49  06/02/77  1336.9 EDT THU
(C)  COPYRIGHT 1976 BY HONEYWELL INFORMATION SYSTEMS INC

ZCRTER  770208
HRS ASSEMBLER 2.49  06/02/77  1934.4 EDT THU
(C)  COPYRIGHT 1976 BY HONEYWELL INFORMATION SYSTEMS INC


**  AC8111         LINK MAP  1901/01/01 0002:44.1
**START   0033
**LOW     0000
**HIGH    0381
**CURRENT 0381

**EXT DEFS
P   7HCOMM  0000
P   7HREL   0000

**  ROOT    0000
 *  AC8111  0000
    AC8111  0033    7CMAIN  0031
 *  ZCRTYU  02B9
    ZCRTY1  02F1    ZCRTY2  0312    ZCRTY3  0334
 *  ZCSTOP  033E
    ZCSTOP  033E
 *  ZCRTER  0341
    ZCRTER  0353

**UNDEF
 *  AC8111  0000
 *  ZCRTYU  02B9
 *  ZCSTOP  033E
 *  ZCRTER  0341

**********
ROOT  AC8111
**********
HIGHEST OVLY    /NUM OF SYMS    1
**********
 SAF
**********
ROOT  AC8111            BASE 0000       ST 0033       -...I HIGH=0381
**********
*SIZE OF ROOT AND STATIC OVLYS= 0381       HI REL RCD=   9
**********
LINK DONE
**********
```

Figure 7-2 (cont). Sample Listings for AC8111

CALLING FORTRAN ROUTINES FROM AN ENTRY-LEVEL COBOL MAIN PROGRAM

Entry-Level COBOL programs can call FORTRAN subroutines and conversely. This enables a COBOL application to utilize the features of the FORTRAN language, such as the intrinsic routines, and FORTRAN run-time libraries.

The COBOL main program must be linked with all the called FORTRAN routines to form one bound unit. The FORTRAN routines and libraries must either be in the working directory or one of the libraries searched by the Linker, as specified by the Linker LIB and LIBn directives.

Figure 7-3 is a sample Entry-Level COBOL source program, COBFRT, whose function is to calculate and print the square roots of three integers. Since the COBOL library does not have a square root routine, a FORTAN subroutine, FRTRAN in Figure 7-4, is used to convert the passed COBOL integer argument values to read values and call the FORTRAN square root routine.

The commands entered from the operator terminal are listed in Figure 7-5. COBFRT.O and FRTRAN.O are both in the working directory FRTCOB, the COBOL run-time library, ZCRT, is in the directory specified by the Linker directive LIB, and the FORTRAN run-time library, ZFRT, is in the directory specified by LIB2. The system volume, ZSYS51, contains the FORTRAN and COBOL compilers, ZFRT, ZCRT and the operating system software. Volume FRTCOB contains the source modules (COBFRT.C and FRTRAN.F), the object modules (COBRFT.O and FRTRAN.O) and the linked bound unit COBFRT.

HONEYWELL INFORMATION SYSTEMS

Technical Publications Remarks Form

CUT ALONG LINE

TITLE	SERIES 60 (LEVEL 6) GCOS 6 MOD 400 PROGRAMMER'S GUIDE ADDENDUM A	ORDER NO.	CB22A, REV. 0
		DATED	JUNE 1978

ERRORS IN PUBLICATION

SUGGESTIONS FOR IMPROVEMENT TO PUBLICATION

Your comments will be promptly investigated by appropriate technical personnel and action will be taken as required. If you require a written reply, check here and furnish complete mailing address below. ☐

FROM: NAME _____ **DATE** _____

TITLE _____

COMPANY _____

ADDRESS _____

_____

PLEASE FOLD AND TAPE –
NOTE: U. S. Postal Service will not deliver stapled forms

FIRST CLASS
PERMIT NO. 39531
WALTHAM, MA
02154

Business Reply Mail
Postage Stamp Not Necessary if Mailed in the United States

Postage Will Be Paid By:

HONEYWELL INFORMATION SYSTEMS
200 SMITH STREET
WALTHAM, MA 02154

ATTENTION: PUBLICATIONS, MS 486

Honeywell

CUT ALONG LINE

FOLD ALONG LINE

FOLD ALONG LINE