CHAPTER 5

THE PAL8 ASSEMBLER

5.1 INTRODUCTION

PAL8, the OS/78 Operating System assembler, generates binary object files from source (ASCII) programs written in the PAL8 assembly language.

PAL8 is a two-pass assembler. During pass 1, the source program is read and an internal symbol table is produced that contains the PAL8 permanent symbols and any new symbols that you define. During pass 2, the assembler reads the source file again, generates the binary code using the symbol table definitions created during pass 1, and continues defining symbols as well. The binary file that is output may be loaded into memory as the "current" executable program by the LOAD command. Absolute binary format consists of 8-bit bytes, containing field setting commands, address setting commands, and sequential data words. An optional third pass will produce a program listing if one is desired. During pass 3, the assembler reads the source file a final time and generates the assembly listing as an ASCII (character string) file. The assembly listing consists of the source statement together with its current location counter and the generated code in octal. The first 40 (decimal) characters of the first line of each page of the listing contain a title, the assembler version number, the date and the listing page number.

Use the OS/78 command PAL to call the assembler. You can also use the commands CREF and EXECUTE as explained in this chapter.

The PAL command specifies the binary and listing output devices and file names, the input devices and file names, and any options that you select. From one to nine input files may be specified. The typical way to assemble, load, and then run a program called PROG is as follows:

```
·PAL PROG          -Assemble the program
·LOAD              -Load the program into memory
·SAVE SYS PROG     -Save the program
·R PROG            -Run the program
```

The long form of the command string is

```
PAL dev:binary,dev:listing,dev:crefls<dev:input,.../options
```

If the extension to the file name is omitted, the following extensions are assumed.

.PA for input files.
.BN for binary output file.
.LS for listing output file.
.TM for intermediate CREF file.

If an assembly or CREF listing is not desired, omit the listing file or CREF file, respectively.

For example, to assemble, load, and run a PAL8 program named SAMPLE, which is stored on diskette unit 1, type

```
     .PAL RXA1:SAMPLE/G-T
```

After assembly the program is loaded and run (since the /G was specified) with the starting address assumed to be location 0200 in field 0; the binary file is stored on the DSK: device as SAMPLE.BN. The -T General-Purpose Dash Option displays the assembled program listing on the terminal (see Table 2-3).

If a binary file is not desired, specify the -NB option at the end of the command line (NB stands for No Binary). For example, to get a listing only, type

```
     .PAL SAMPLE-LS-NB
```

The -LS option indicates that a listing should be produced.

The assembler displays any error messages encountered in the program on the terminal, even when a listing is not produced. Typing CTRL/O at the keyboard during an assembly suppresses the display of error messages. However, messages are still printed in the listing file (if any) and occur immediately before the line that is in error.

For example, the command line

```
     .PAL SAMPLE/S-LS
```

causes PAL8 to assemble SAMPLE.PA (or SAMPLE), generating DSK:SAMPLE.BN and putting the listing into the file SAMPLE.LS on the default device DSK. The /S option suppresses listing of the symbol table.

The command line

```
     .PAL BIN<SAMPLE.PA/G=600
```

assembles SAMPLE.PA, creates a binary output file named BIN.BN, loads the file BIN.BN, and starts it at location 600. The construction =600 is an option that specifies the starting address.

Assembly can be terminated at any time by typing CTRL/C on the keyboard, and any output files being stored will be deleted. Otherwise, PAL8 always returns to the monitor upon completion of assembly.

A source program may consist of a number of source file modules to be assembled together. You do this by specifying a string of input device and file names separated by commas. For example,

```
     .PAL PART1,RXA1:PART2,RL0A:PART3
```

assembles a three-part program. This technique is useful when it is desired to assemble two programs that are identical except for a few lines at the beginning of the programs. Different lines can be broken out into a "prefix file". For example, two different file assemblies may be generated by

```
.PAL PRFX1,FILE
```

and

```
.PAL PRFX2,FILE
```

You can enter up to nine input files to be treated as one source input in a command line.

If more than one input file is specified, and output files are desired but not explicitly specified, the name of the first input file is used for the output file names. For example,

```
.PAL A,B
```

produces the binary file A.BN.

If a file name other than the first input file is desired for the binary name, use the -NB General-Purpose Dash Option after the last input file name not desired as the binary file name. For example,

```
.PAL A-NB,B
```

produces DSK:B.BN and

```
.PAL A,B,C-NB,D,E,F
```

produces DSK:D.BN.

If a -LS option is specified, it must appear immediately after an input file name. This is the name that will be used for the name of the listing file. For example,

```
.PAL A,B-LS
```

produces DSK:B.LS while

```
.PAL A-LS,B
```

produces DSK:A.LS

The -L or -T General-Purpose Dash Options used with a PAL or COMPILE command send the listing output file to the line printer and terminal respectively.

Note that the PAL command normally produces a binary file even when a name is not given. Thus, typing

```
.PAL ,LPT:<file
```

produces a binary file.

If you do not specify an extension, PAL assumes that the input file extension is .PA. Thus, the command

```
.PAL TEST
```

causes the assembler to search for a file namel DSK:TEST.PA. If no file with .PA is found on DSK:, the assembler then searches for a file named TEST with no extension. It is good practice when creating a PAL8 source file to include a .PA extension to remind you what type of source file it is.

The COMPILE and EXECUTE commands may also be used to invoke PAL8. These commands search the directory of the specified device for the file given with the command, and if one is found with a .PA extension, PAL8 is invoked. For example,

```
     .COMPILE TEST
```

will run PAL8 if TEST.PA is found. An unusual extension may be explicitly specified by typing

```
     .PAL TEST.XX
```

which will assemble DSK:TEST.XX. To specify PAL8 as the processor in the COMPILE command, use the -PA General-Purpose Dash Option in the command line as follows:

```
     .COMPILE TEST.XX-PA
```

The EXECUTE command is similar to the COMPILE command except that the EXECUTE command is supported by the /G option.

If an argument is not given with a PAL or COMPILE or EXECUTE command, the argument used with the last such command is assumed when that command is used again.

5.2 CREATING AND RUNNING A PAL8 PROGRAM

The following steps demonstrate the procedure for creating and running a PAL8 program.

5.2.1 Creating a Program

Create the assembly language source file by calling the Editor as follows:

```
     .CREATE SAMPLE.PA
```

Since a new program is being created, only a single file name need be specified. The OS/78 Editor will then display a number sign (#) to indicate it is ready to accept a command. (See Chapter 4 for a detailed discussion of the OS/78 Editor.)

Type the A (Append) command to allow the Editor to accept text. Then type in the program, one line at a time. Press the RETURN key after each line.

```
#A

/ROUTINE TO TYPE A MESSAGE
        *200
        MONADR=7600
START,  CLA CLL             /CLEAR ACCUMULATOR AND LINK
        TLS                 /CLEAR TERMINAL FLAG
        TAD BUFADR          /SET UP POINTER
        DCA PNTR            /FOR GETTING CHARACTERS
```

```
NEXT,    TSF                  /SKIP IF TERMINAL FLAG SET
         JMP .-1              /NO: CHECK AGAIN
         TAD I PNTR           /GET A CHARACTER
         TLS                  /PRINT A CHARACTER
         ISZ PNTR             /DONE YET?
         CLA CLL              /CLEAR ACCUMULATOR AND LINK
         TAD I PNTR           /GET ANOTHER CHARACTER
         SZA CLA              /JUMP ON ZERO AND CLEAR
         JMP NEXT             /GET READY TO PRINT ANOTHER
         JMP I MON            /RETURN TO MONITOR
BUFADR,  BUFF                 /BUFFER ADDRESS
PNTR,    BUFF                 /POINTER
BUFF,    215;212;"H;"E;"L;"L;"O;"!;0
MON,     MONADR               /MONITOR ENTRY POINT
```

Now type a CTRL/L to terminate input. This command returns you to the Editor command mode.

Type the L (List) command in response to the Editor's number sign (#) to list the text that was inserted into the text buffer.

When you are satisfied that the input is correct, type the E (Exit) command to store the file and return to the monitor.

5.2.2 Assembling a Program

Now assemble the source program just created. Use the command:

```
.PAL SAMPLE-LS
```

This command creates two files, a binary file called SAMPLE.BN, and a listing file (-LS option) called SAMPLE.LS. Use the TYPE command to display the listing on the terminal or the LIST command to print the listing on a line printer.

The assembly listing produced by PAL appears as follows:

```
/ROUTINE TO TYPE A MESSAGE        PAL8-V13A  14-MARCH-79 PAGE 1

                    /ROUTINE TO TYPE A MESSAGE
        0200                *200
        7600                MONADR=7600
000200  7300  START,  CLA CLL           /CLEAR ACCUMULATOR AND LINK
000201  6046          TLS               /CLEAR TERMINAL FLAG
000202  1216          TAD BUFADR        /SET UP POINTER
000203  3217          DCA PNTR          /FOR GETTING CHARACTERS
000204  6041  NEXT,   TSF               /SKIP IF TERMINAL FLAG SET
000205  5204          JMP .-1           /NO: CHECK AGAIN
000206  1617          TAD I PNTR        /GET A CHARACTER
000207  6046          TLS               /PRINT A CHARACTER
000210  2217          ISZ PNTR          /DONE YET?
000211  7300          CLA CLL           /CLEAR ACCUMULATOR AND LINK
000212  1617          TAD I PNTR        /GET ANOTHER CHARACTER
000213  7640          SZA CLA           /JUMP ON ZERO AND CLEAR
000214  5204          JMP NEXT          /GET READY TO PRINT ANOTHER
000215  5631          JMP I MON         /RETURN TO MONITOR
000216  0220  BUFADR, BUFF    /BUFFER ADDRESS
000217  0220  PNTR,   BUFF    /POINTER
000220  0215  BUFF,   215;212;"H;"E;"L;"L;"O;"!;0
```