16.5 Other Overlay Systems

You can also use LINK-80 to produce a system of overlays that is not a tree structure, but contains instead a number of separate overlay areas, as shown in Figure 16-2.

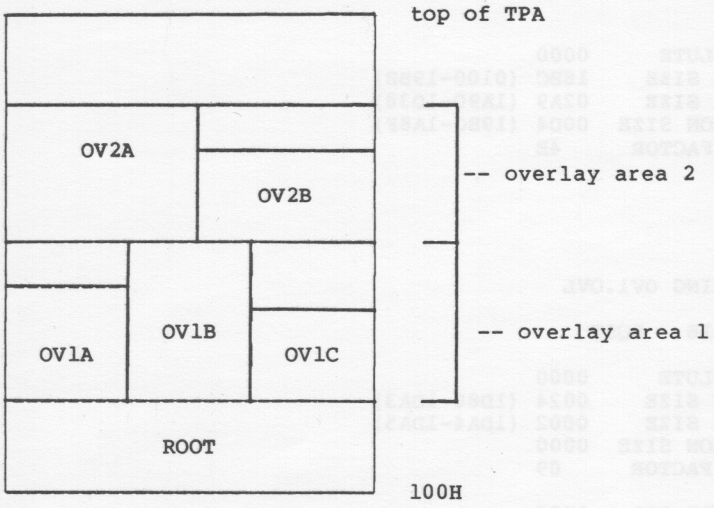

Figure 16-2. Separate Overlay System

In such a system, the root module can reference any of the overlays. An overlay can reference entry points in the root module or the main entry point of any overlay that is not in the same overlay area.

Linking a system of overlays as shown above is done in a number of steps. One link operation must be performed for each overlay area because LINK-80 must be supplied the address of the top of the overlay area when linking the next higher overlay area.

For example, from the command

```
A>LINK ROOT (OV1A)(OV1B)(OV1C)
```

LINK-80 generates the three overlays in overlay area 1 and indicates the top address of the module. This address is then supplied as the load address in the next command:

```
A>LINK ROOT (OV2A[Lmod top]) (OV2B [Lmod top])
```

This command creates the overlays for overlay area 2 at the appropriate address. Note that the overlay area that is the highest in memory should be linked last because LINK-80 always writes the module top address into the root module at the end of the link operation.

At some point after the entire system has been linked, it is desirable to relink only one overlay, which might not be at the top overlay area. This can be done using the $OZ switch to prevent generation of a root module that would contain an erroneous ?MEMRY value.

If only OV1C is changed, the following command creates a new OV1C overlay without creating a new root module. The root module is included in the LINK command so that LINK-80 can resolve references to the root from OV1C.

For example,

```
A>LINK ROOT [$OZ](OV1C[$OA])
```

Note: when using this type of overlay system, you must ensure that none of the overlays overlap and that no overlay attempts to reference another overlay in the same overlay area.

End of Section 16

Section 17
LIB-80

17.1 Introduction

LIB-80 is a utility program that creates libraries. Libraries are files consisting of any number of relocatable object modules. LIB-80 can perform the following functions:

- o concatenate a group of REL files into a library
- o create an indexed library (IRL)
- o select, delete, or replace modules from a library
- o print module names and PUBLICS from a library

17.2 LIB-80 Operation

LIB-80 takes the general command form:

```
A>LIB filename=filename1,...,filenameN
```

This command creates a library called filename.REL from the files filename1.REL,...,filenameN.REL. If you omit the filetypes, LIB-80 assumes filetype REL.

A filename can be followed by a group of module names enclosed in parentheses. Only the modules indicated are included in the LIB function being performed. If omitted, LIB-80 includes all the modules in the file.

For example, the command

```
A>LIB TEST=A(A1,A2),B,C(C1-C4,C6)
```

creates a file named TEST.REL consisting of the modules A1 and A2 from A.REL, all the modules from B.REL, and the modules between C1 and C4, and C6 from C.REL.

LIB-80 can delete or replace modules in a library with a single command. To do this, enter the names of the modules to be affected and enclose them in angle brackets immediately following the name of the source file that contains the modules.

For example, the command

```
A>LIB NEWLIB=OLDLIB<MOD1>
```

creates a new library named NEWLIB.REL that is the same as OLDLIB.REL except that the module MOD1 is replaced with the file MOD1.REL. Use this form of the command if the name of the module being replaced is the same as the filename of the REL file replacing the module.

The command form:

LIB NEWLIB=OLDLIB<MOD1=FILE1>

creates a new library with the module MOD1 replaced by the file FILE1.REL. Use this form of the command when the name of the module being replaced is not the same as the name of the file replacing it. This form of the command must be used if the filename within angle brackets has more than 6 characters because module names in the REL file are truncated to 6 characters.

The command form

LIB NEWLIB=OLDLIB<MOD1>

creates a new library from OLDLIB.REL, deleting the module MOD1.

The command form

LIB NEWLIB=OLDLIB<MOD1,MOD2=FILE2,MOD3=>

creates a new library from OLDLIB.REL with MOD1.REL replacing the module MOD1, FILE2.REL replacing MOD2, and deleting MOD3. This command demonstrates that a number of replace and/or delete instructions can be included within the angle brackets.

17.3 LIB-80 Switches

LIB-80 supports optional parameters in the command line that control its operation. These parameters are called switches. They are enclosed in square brackets and appear after the first filename in the LIB command. Table 17-1 shows the LIB-80 switches.

Table 17-1. LIB-80 Switches

Switch	Function
D	displays contents of object modules in ASCII form.
I	creates an indexed library (IRL).
M	prints module names.
P	prints module names and PUBLICS.

For example, the command

A>LIB TEST=A,B,C

creates a file TEST.REL consisting of A.REL, B.REL, and C.REL.

The command

```
A>LIB TEST=TEST,D
```

appends D.REL to the end of TEST.REL.

The command

```
A>LIB TEST[I]
```

creates an indexed library TEST.IRL from TEST.REL.

The command

```
A>LIB TEST[I]=A,B,C,D
```

performs the same function as the preceding examples, but LIB-80 creates a file TEST.IRL without creating a file TEST.REL.

The command

```
A>LIB TEST [P]
```

lists all the module names and PUBLICS in TEST.REL.

End of Section 17

Appendix A
MAC/RMAC Error Messages

When errors occur within the assembly language program, they are listed as single-character flags in the leftmost position of the source listing. The line in error is also echoed at the console so that the .PRN file need not be examined to determine if errors are present. The single-character error codes are listed in Table A-1.

Table A-1. MAC/RMAC Error Messages

Flag	Meaning
B	Balance error: macro does not terminate properly, or conditional assembly operation is ill formed.
C	Comma error: expression was encountered but not delimited properly from the next item by a comma.
D	Data error: element in a data statement (DB or DW) cannot be placed in the specified data area.
E	Expression error: expression is ill formed and cannot be computed at assembly time.
I	Invalid character error: a nongraphic character has been found in the line other than a carriage return, line-feed, tab, or end-of-file; edit the file, delete the line with the I error, and retype the line.
L	Label error: label cannot appear in this context; it might be a duplicate label.
M	Macro overflow error: internal macro expansion table overflow; might be due to too many nested invocations or infinite recursion.
N	Not implemented error: features that appear in RMAC, such as relocation, are recognized, but flagged in MAC.
O	Overflow error: expression is too complicated (i.e., has too many pending operators), string is too long, or too many successive substitutions of a formal parameter by its actual value in a macro expansion. This error also occurs if the number of LOCAL labels exceeds 9999.

Table A-1. (continued)

Flag	Meaning
P	Phase error: label does not have the same value on the two passes through the program, or the order of macro definition differs between the two successive passes; might be due to MACLIB that follows a mainline macro; if so, move the MACLIB to the top of the program.
R	Register error: the value specified as a register is not compatible with the operation code.
S	Syntax error: the fields of this statement are ill formed and cannot be processed properly; might be due to invalid characters or delimiters that are out of place.
U	Undefined symbol: a label operand in this statement has not been defined elsewhere in the program.
V	Value error: operand encountered in an expression is improperly formed; might be due to delimiter out of place or nonnumeric operand.

The error messages shown in Table A-2 indicate terminal error conditions that abort the MAC execution. Whenever possible, the disk drive name, followed by the relevant filename, is printed with the message.

Table A-2. Terminal Error Conditions

Message	Meaning
CANNOT CLOSE FILE:	An output file cannot be closed. The disk might be write protected.
INVALID PARAMETER:	An invalid assembly parameter was found in the input line. The assembly parameters are printed at the console up to the point of the error.

Table A-2. (continued)

Message	Meaning
NO DIRECTORY SPACE:	The disk directory is full. Use the ERA command of the CCP to remove files you do not need. Often superfluous .HEX, .PRN, and .SYM files can be removed.
NO SOURCE FILE PRESENT:	The source program file (.ASM) following the MAC command cannot be found on the specified disk. Use the DIR command in the CCP to locate the source file.
OUTPUT FILE READ ERROR:	An output file cannot be written properly, probably due to a full disk. As in the NO DIRECTORY SPACE error above, use the CCP commands to erase unnecessary files from disk.
SOURCE FILENAME ERROR:	The form of the source filename is invalid or not specified. The command form must be MAC filename $assembly parameters where the filename is the primary name (up to eight characters) of the source file, with an assumed filetype of .ASM. Filetype is not specified.
SOURCE FILE READ ERROR:	The source file cannot be read properly by the macro assembler. Use the CCP TYPE command to display the file contents at the console.

Table A-2. (continued)

Message	Meaning
UNBALANCED MACRO LIBRARY:	A MACRO definition was started within a macro library, but the end of file was found in the library before the balancing ENDM was encountered. Examine the macro library using the TYPE command of the CCP, or use the +L assembly parameter to ensure that the library is properly balanced.

End of Appendix A

Appendix B
XREF Error Messages

During the course of operation, XREF might display error messages. These error messages and brief explanations of their causes are shown in Table B-1.

Table B-1. XREF Error Messages

Error	Cause
No SYM file	The file filename.SYM is not present on the default or specified drive.
No PRN file	The file filename.PRN is not present on the default or specified drive.
Symbol Table overflow	No space is available for an attempted symbol allocation.
Invalid SYM file format	XREF issues this message when it reads an invalid filename.SYM file. Specifically, a line in the SYM file that does not terminate with a CRLF forces this error message.
Symbol Table reference overflow	No space is available for an attempted symbol reference allocation.
filename.XRF make error	XREF issues this message if the CP/M BDOS returns an error code after a make file request for the file filename.XRF. This error code usually indicates that no directory space exists on the default or specified drive.

Table 15-1. (continued)

Error	Cause
filename.XRF close error	XREF issues this message if the CP/M BDOS returns an error code after a close request for the file filename.XRF.
filename.XRF write error	XREF issues this message if the CP/M BDOS returns an error code after a write request for the file filename.XRF. This error code usually indicates that no unallocated data blocks are available, or no directory space exists on the default or specified drive.

End of Appendix B

Appendix C
LINK-80 Error Messages

When LINK-80 detects any kind of command line error, it echoes the command tail up to the point where the error occurs and follows it with a question mark. For example,

```
A>link a, b, c; d
A, B, C;?

A>link longfilename
LONGFILEN?
```

During the course of operation, LINK-80 can display error messages. These error messages are described in Table C-1 below.

Table C-1. LINK-80 Error Messages

Message	Meaning
CANNOT CLOSE:	An output file cannot be closed. The disk might be write-protected.
COMMON ERROR:	An undefined common block has been selected.
DIRECTORY FULL:	There is no directory space for the output files or intermediate files.
DISK READ ERROR:	A file cannot be read properly.
DISK WRITE ERROR:	A file cannot be written properly, probably because the disk is full.

Table C-1. (continued)

Message	Meaning
FIRST COMMON NOT LARGEST:	A subsequent COMMON declaration is larger than the first COMMON declaration for the indicated block. Check that the files being linked are in the proper order, or that the modules in a library are in the proper order.
INDEX ERROR:	The index of an IRL file contains invalid information.
INSUFFICIENT MEMORY:	There is not enough memory for LINK-80 to allocate its buffers. Try using the A switch.
INVALID REL FILE:	The file indicated contains an invalid bit pattern. Make sure that a REL or IRL file has been specified.
MAIN MODULE ERROR:	A second main module was encountered.
MEMORY OVERFLOW:	There is not enough memory to complete the link operation. Try using the A switch.
MULTIPLE DEFINITION:	The specified symbol is defined in more than one of the modules being linked.

Table C-1. (continued)

Message	Meaning
NO FILE:	The indicated file cannot be found.
OVERLAPPING SEGMENTS:	LINK-80 attempted to write a segment into memory already used by another segment. Probably caused by incorrect use of P and/or D switches.
UNDEFINED START SYMBOL:	The symbol specified with the G switch is not defined in any of the modules being linked.
UNDEFINED SYMBOLS:	The symbols following this message are referenced but not defined in any of the modules being linked.
UNRECOGNIZED ITEM:	An unfamiliar bit pattern has been scanned and ignored by LINK-80.

End of Appendix C

Appendix D
Overlay Manager Run-time Error Messages

At run-time, the Overlay Manager can display certain error messages. These messages and a brief explanation of their causes are shown in Table D-1.

Table D-1. Run-time Error Messages

Error	Cause
ERROR (8) OVERLAY, NO FILE d:filename.OVL	The Overlay Manager cannot find the indicated file.
ERROR (9) OVERLAY, DRIVE d:filename.OVL	An invalid drive code was passed as a parameter to ?ovlay.
ERROR (10) OVERLAY, SIZE d:filename.OVL	The indicated overlay would overwrite the PL/I stack and/or free space if it were loaded.
ERROR (11) OVERLAY, NESTING d:filename.OVL	Loading the indicated overlay would exceed the maximum nesting depth.
ERROR (12) OVERLAY, READ d:filename.OVL	Disk read error during overlay load, probably caused by premature EOF.

End of Appendix D

Appendix E
LIB-80 Error Messages

During the course of operation, LIB-80 can display error messages. These error messages and a brief explanation of their causes are given in Table E-1.

Table E-1. LIB-80 Error Messages

Error	Cause
CANNOT CLOSE:	LIB-80 cannot close the output file. The disk might be write-protected.
DIRECTORY FULL:	There is no directory space for the output file.
DISK READ ERROR:	LIB-80 cannot read the file properly.
DISK WRITE ERROR:	LIB-80 cannot write to the file properly, probably due to a full disk.
FILE NAME ERROR:	The form of a source filename is invalid.
NO FILE:	LIB-80 cannot find the indicated file.
NO MODULE:	LIB-80 cannot find the indicated module.
SYNTAX ERROR:	The LIB-80 command line is not properly formed.

End of Appendix E

End of Appendix E

Appendix F
8080 CPU Instructions

OP CODE	MNEMONIC	OP CODE	MNEMONIC	OP CODE	MNEMONIC
00	NOP	2B	DCX H	56	MOV D,M
01	LXI B,D16	2C	INR L	57	MOV D,A
02	STAX B	2D	DCR L	58	MOV E,B
03	INX B	2E	MVI L,D8	59	MOV E,C
04	INR B	2F	CMA	5A	MOV E,D
05	DCR B	30	---	5B	MOV E,E
06	MVI B,D8	31	LXI SP,D16	5C	MOV E,H
07	RLC	32	STA Adr	5D	MOV E,L
08	---	33	INX SP	5E	MOV E,M
09	DAD B	34	INR M	5F	MOV E,A
0A	LDAX B	35	DCR M	60	MOV H,B
0B	DCX B	36	MVI M,D8	61	MOV H,C
0C	INR C	37	STC	62	MOV H,D
0D	DCR C	38	---	63	MOV H,E
0E	MVI C,D8	39	DAD SP	64	MOV H,H
0F	RRC	3A	LDA Adr	65	MOV H,L
10	---	3B	DCX SP	66	MOV H,M
11	LXI D,D16	3C	INR A	67	MOV H,A
12	STAX D	3D	DCR A	68	MOV L,B
13	INX D	3E	MVI A,D8	69	MOV L,C
14	INR D	3F	CMC	6A	MOV L,D
15	DCR D	40	MOV B,B	6B	MOV L,E
16	MVI D,D8	41	MOV B,C	6C	MOV L,H
17	RAL	42	MOV B,D	6D	MOV L,L
18	---	43	MOV B,E	6E	MOV L,M
19	DAD D	44	MOV B,H	6F	MOV L,A
1A	LDAX D	45	MOV B,L	70	MOV M,B
1B	DCX D	46	MOV B,M	71	MOV M,C
1C	INR E	47	MOV B,A	72	MOV M,D
1D	DCR E	48	MOV C,B	73	MOV M,E
1E	MVI E,D8	49	MOV C,C	74	MOV M,H
1F	RAR	4A	MOV C,D	75	MOV M,L
20	---	4B	MOV C,E	76	HLT
21	LXI H,D16	4C	MOV C,H	77	MOV M,A
22	SHLD Adr	4D	MOV C,L	78	MOV A,B
23	INX H	4E	MOV C,M	79	MOV A,C
24	INR H	4F	MOV C,A	7A	MOV A,D
25	DCR H	50	MOV D,B	7B	MOV A,E
26	MVI H,D8	51	MOV D,C	7C	MOV A,H
27	DAA	52	MOV D,D	7D	MOV A,L
28	---	53	MOV D,E	7E	MOV A,M
29	DAD H	54	MOV D,H	7F	MOV A,A
2A	LHLD Adr	55	MOV D,L	80	ADD B
81	ADD C	AC	XRA H	D7	RST 2
82	ADD D	AD	XRA L	D8	RC
83	ADD E	AE	XRA M	D9	---
84	ADD H	AF	XRA A	DA	JC Adr
85	ADD L	B0	ORA B	DB	IN D8

8080 CPU Instructions in Operation Code Sequence (continued)

OP CODE	MNEMONIC		OP CODE	MNEMONIC		OP CODE	MNEMONIC	
86	ADD	M	B1	ORA	C	DC	CC	Adr
87	ADD	A	B2	ORA	D	DD	---	
88	ADC	B	B3	ORA	E	DE	SBI	D8
89	ADC	C	B4	ORA	H	DF	RST	3
8A	ADC	D	B5	ORA	L	E0	RPO	
8B	ADC	E	B6	ORA	M	E1	POP	H
8C	ADC	H	B7	ORA	A	E2	JPO	Adr
8D	ADC	L	B8	CMP	B	E3	XTHL	
8E	ADC	M	B9	CMP	C	E4	CPO	Adr
8F	ADC	A	BA	CMP	D	E5	PUSH	H
90	SUB	B	BB	CMP	E	E6	ANI	D8
91	SUB	C	BC	CMP	H	E7	RST	4
92	SUB	D	BD	CMP	L	E8	RPE	
93	SUB	E	BE	CMP	M	E9	PCHL	
94	SUB	H	BF	CMP	A	EA	JPE	Adr
95	SUB	L	C0	RNZ		EB	XCHG	
96	SUB	M	C1	POP	B	EC	CPE	Adr
97	SUB	A	C2	JNZ	Adr	ED	---	
98	SBB	B	C3	JMP	Adr	EE	XRI	D8
99	SBB	C	C4	CNZ	Adr	EF	RST	5
9A	SBB	D	C5	PUSH	B	F0	RP	
9B	SBB	E	C6	ADI	D8	F1	POP	PSW
9C	SBB	H	C7	RST	0	F2	JP	Adr
9D	SBB	L	C8	RZ		F3	DI	
9E	SBB	M	C9	RET	Adr	F4	CP	Adr
9F	SBB	A	CA	JZ		F5	PUSH	PSW
A0	ANA	B	CB	---		F6	ORI	D8
A1	ANA	C	CC	CZ	Adr	F7	RST	6
A2	ANA	D	CD	CALL	Adr	F8	RM	
A3	ANA	E	CE	ACI	D8	F9	SPHL	
A4	ANA	H	CF	RST	1	FA	JM	Adr
A5	ANA	L	D0	RNC		FB	EI	
A6	ANA	M	D1	POP	D	FC	CM	Adr
A7	ANA	A	D2	JNC	Adr	FD	---	
A8	XRA	B	D3	OUT	D8	FE	CPI	D8
A9	XRA	C	D4	CNC	Adr	FF	RST	7
AA	XRA	D	D5	PUSH	D			
AB	XRA	E	D6	SUI	D8			

D8 = constant or logical/arithmetic expression that evaluates to an 8 bit quantity.
Adr = 16-bit address.
D16 = constant or logical/arithmetic expression that evaluates to a 16 bit data quantity.

End of Appendix F