Microsoft LINK-80 LOADER

CP/M® Version

Software Reference Manual

for HEATH/ZENITH 8-bit digital computer systems

Table of Contents

LINK-80, Linking Loader

LINK-80 Linking Loader

OVERVIEW

The following Section contains reference information about the LINK-80 Linking Loader. The Linking Loader is used to load the relocatable modules produced by the FORTRAN-80, COBOL-80 or BASIC compiler and the Macro-80 Assembler. The Linking Loader also links these modules to any internal routines that may be needed for execution of the relocatable module.

For example, to perform formatted random I/O several routines are referenced in the FORTRAN-80 library. These routines contain the actual machine language code needed in order to access the disk drive. The linker is used to link the main program to these routines.

The linker can also be used to create an absolute file that can be executed under CP/M. This file has the default extension .COM and is completely compatable with CP/M.

NOTE: Be sure to use only 8080 op-codes if the absolute file is intended to run on an H8 without the HA-8-6 Z80 CPU board.

LINK-80 COMMAND STRINGS

To run LINK-80, type L80 followed by a carriage return. LINK-80 will return the prompt "*". Each command to LINK-80 consists of a string of file names and switches separated by commas:

objdev1:filename.ext/switch1,objdev2:filename.ext,...

If the input device for a file is omitted, the default drive is used. If the extension of an input file is omitted, the default is .REL. After each line is typed, LINK-80 will load or search (see /S below) the specified files. After LINK-80 finishes this process, it will list all symbols that remained undefined followed by an asterisk.

Before execution begins, LINK-80 will always search the system library to satisfy any unresolved external references. The system library must reside on the default drive.

If the user wishes to first search non-standard libraries, the file names that are followed by /S should be appended to the end of the loader command string.

The following examples illustrate a typical use of the Linking Loader.

```
*TEST
```

This will load the file TEST.REL.

```
*TEST/N/E
```

This command string tells the linker to output the results of the linking and loading process in a file called TEST.COM. The /E will cause the linker to first search the system library to clear up any unresolved references, then exit to CP/M.

LINK-80 Switches

A number of switches may be given in the LINK-80 command string to specify actions affecting the loading process. Each switch must be preceded by a slash (/).

These switches are:

/R

Reset. Put loader back in its initial state. Use /R if the wrong file is accessed and it is necessary to re-start. /R takes effect as soon as it is encountered in a command string.

/E or /E:Name

Exit LINK-80 and return to CP/M. The system library will be searched on the default drive to satisfy any existing undefined globals.

The optional form /E:Name (where Name is a global symbol previously defined in one of the modules) uses Name for the start address of the program.

/G or /G:Name

Start execution of the program as soon as the current command line has been interpreted. The system library will be searched on the default disk to satisfy any existing undefined globals if they exist.

Before execution actually begins, LINK-80 prints two numbers and a BEGIN EXECUTION message. The two numbers are the start address, and the address of the next available byte.

The optional form /G:Name (where Name is a global symbol previously defined in one of the modules) uses Name for the start address of the program.

/N

If a <filename>/N is specified, the program will be saved on disk under the selected name (with a default extension of .COM when a /E or /G is done. A jump to the start of the program is inserted so the program will run properly.

/P and /D

/P and /D allow the origin(s) to be set for the next program loaded. /P and /D take effect when seen and they have no effect on programs already loaded. The form is /P:<address> or /D:<address>, where <address> is the desired origin in the current radix. (Default radix is hex. /O sets radix to octal; /H to hex.)

Do not use /P or /D to load programs or data into the locations of the loader's jump to the start address (100H to 102H) unless it is to load the start of the program there. If programs or data are loaded into these locations, the jump will not be generated.

If no /D is given, data areas are loaded before program areas for each module. If a /D is given, all Data and Common areas are loaded starting at the data origin and the program area at the program origin.

/U

List the origin and end of the program and data area and all undefined globals as soon as the current command line has been interpreted. The program information is only printed if a /D has been done.

/M

List the origin and end of the program and data area, all defined globals and their values, and all undefined globals followed by an asterisk. The program information is only printed if a /D has been done.

/S
Search the filename immediately preceding the /S in the command string to satisfy any undefined globals.

/X

If a filename/N was specified, /X will cause the file to be saved in Intel ASCII HEX format with an extension of HEX.

> EXAMPLE: FOO/N/X/E will create an Intel ASCII HEX formatted load module named FOO.HEX.

/Y

If a filename/N was specified, /Y will create a filename.SYM file when /E is entered. This file contains the names and addresses of all Globals for use with Digital Research's Symbolic Debugger, SID and ZSID.

> EXAMPLE: FOO/N/Y/E Creates FOO.COM and FOO.SYM. MYPROG/N/X/Y/E creates MYPROG.HEX and MYPROG.SYM.

LINK-80 Error Messages

LINK-80 has the following error messages:

?No Start Address

A /G switch was issued, but no main program had been loaded.

?Loading Error

The last file given for input was not a properly formatted LINK-80 object file.

?Out of Memory

Not enough memory to load program. (A minimum of 40K RAM is required.)

?Command Error

Unrecognizable LINK-80 command string.

?<file> Not Found

<file>, as given in the command string, did not exist.

%2nd COMMON Larger /XXXXXX/

The first definition of COMMON block /XXXXXX/ was not the largest definition. Re-order module loading sequence or change COMMON block definitions. (See the FORTRAN Reference Manual for more information on the COMMON statement.)

%Mult. Def. Global YYYYYY

More than one definition for the global (internal) symbol YYYYYY was encountered during the loading process.

%Overlaying Program Area

A /D or /P will cause already loaded data to be destroyed.

?Intersecting Program Area
Data

The program and data area intersect and an address or external chain entry is in this intersection. The final value cannot be converted to a current value since it is in the area intersection.

?Start Symbol - <name> - Undefined

After a /E: or /G: is given, the symbol specified was not defined.

Origin Above Loader Memory, Move Anyway (Y or N)?
Below

After a /E or /G was given, either the data or program area has an origin or top which lies outside loader memory. If a Y <cr> is given, LINK-80 will move the area and continue. If anything else is given, LINK-80 will exit.

In either case, if a /N was given, the image will already have been saved.

?Can't Save Object File

A disk error occurred when the file was being saved. Usually this occurs when there is no more room left on the disk.

?Nothing Loaded

A <filename>/S or /E or /G was given but no object file was loaded. That is, an attempt was made to search a library, exit the Linker, or execute a program, when in fact nothing had been loaded. For example:

TEST/N/E Results in '?Nothing Loaded' because TEST/N names TEST.COM, but does not load TEST.REL.

FORMAT OF LINK-80 COMPATIBLE OBJECT FILES

The following information is reference material for users who wish to know the load format of LINK-80 relocatable object files.

LINK-compatible object files consist of a bit stream. Individual fields within the bit stream are not aligned on byte boundaries, except as noted below. Use of a bit stream for relocatable object files keeps the size of object files to a minimum, thereby decreasing the number of disk reads/writes.

There are two basic types of load items: Absolute and Relocatable. The first bit of an item indicates one of these two types. If the first bit is a 0, the following 8 bits are loaded as an absolute byte. If the first bit is a 1, the next 2 bits are used to indicate one of four types of relocatable items:

00 Special LINK item (see below).

01 Program Relative. Load the following 16 bits after adding the current Program base.

10 Data Relative. Load the following 16 bits after adding the current Data base.

11 Common Relative. Load the following 16 bits after adding the current Common base.

Special LINK items consist of the bit stream 100 followed by:

A four-bit control field.

An optional A field consisting of a two-bit address type that is the same as the two-bit field above except 00 specifies absolute address.

An optional B field consisting of 3 bits that give a symbol length and up to 8 bits for each character of the symbol.

A general representation of a special LINK item is:

1 00 xxxx yy nn zzz + characters of symbol name
(yy nn: A field; zzz + characters of symbol name: B field)

xxxx	Four-bit control field (0-15 below)
yy	Two-bit address type field
nn	Sixteen-bit value
zzz	Three-bit symbol length field

The following special types have a B-field only:

0 Entry symbol (name for search)
1 Select COMMON block
2 Program name
3 Request library search
4 Reserved for future expansion

The following special LINK items have both an A field and a B field:

5 Define COMMON size
6 Chain external (A is head of address chain, B is name of external symbol)
7 Define entry point (A is address, B is name)
8 External-offset. Used for JMP and CALL to externals.

The following special LINK items have an A field only:

9 External + offset. The A value will be added to the two bytes starting at the current location counter immediately before execution.
10 Define size of Data area (A is size)
11 Set loading location counter to A
12 Chain address. A is head of chain, replace all entries in chain with current location counter. The last entry in the chain has an address field of absolute zero.
13 Define program size (A is size)
14 End program (forces to byte boundary)

The following special Link item has neither an A nor a B field:

15 End File

LINK-80 Linking Loader Index