This manual describes the Lingage Editor included in the development package available as an option with the OASIS Operating System.

This manual, named LINK, like all OASIS documentation manuals, has the manual name and revision number (if applicable) in the lower, inside corner of each page of the body of the manual. In most chapters of the manual the last primary subject being discussed on a page will be identified in the lower outside corner of the page.

Related Documentation

The following publications provide additional information useful in the use of this program:

OASIS System Reference Manual
OASIS EXEC Language Reference Manual
OASIS Text Editor Reference Manual
OASIS MACRO Assembler Language Reference Manual
OASIS Dynamic Debugger Reference Manual
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAPTER 1 INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>CHAPTER 2 LINK COMMAND</td>
<td>2</td>
</tr>
<tr>
<td>CHAPTER 3 LINK INPUT FILES</td>
<td>3</td>
</tr>
<tr>
<td>3.1 Specification File</td>
<td>3</td>
</tr>
<tr>
<td>3.1.1 DEFINE Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.2 END Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.3 ENTRY Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.4 INCLUDE Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.5 IGNORE Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.6 LIST Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.7 NAME Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.8 ORIGIN Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.9 OVERLAY Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.10 REPLACE Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.11 SET Command</td>
<td>3</td>
</tr>
<tr>
<td>3.1.12 Comments</td>
<td>3</td>
</tr>
<tr>
<td>3.2 Object File</td>
<td>6</td>
</tr>
<tr>
<td>3.2.1 PAB Definition Record (P)</td>
<td>6</td>
</tr>
<tr>
<td>3.2.2 Text Record (T)</td>
<td>6</td>
</tr>
<tr>
<td>3.2.3 Relocation Record (R)</td>
<td>6</td>
</tr>
<tr>
<td>3.2.4 Entry Definition Record (E)</td>
<td>6</td>
</tr>
<tr>
<td>3.2.5 External Reference Record (X)</td>
<td>7</td>
</tr>
<tr>
<td>3.2.6 PAB to PAB Reference Record (F)</td>
<td>7</td>
</tr>
<tr>
<td>3.2.7 End of File Record (Z)</td>
<td>7</td>
</tr>
<tr>
<td>3.2.8 Object File Record Examples</td>
<td>7</td>
</tr>
<tr>
<td>CHAPTER 4 LINK OUTPUT FILES</td>
<td>8</td>
</tr>
<tr>
<td>4.1 Absolute Load Module File</td>
<td>8</td>
</tr>
<tr>
<td>4.2 Relocatable Load Module File</td>
<td>8</td>
</tr>
<tr>
<td>4.3 Map Listing File</td>
<td>8</td>
</tr>
<tr>
<td>APPENDIX A LINK EXAMPLES</td>
<td>10</td>
</tr>
<tr>
<td>A.1 Example One: Simple, Single PAB</td>
<td>10</td>
</tr>
<tr>
<td>A.2 Example Two: Specification File</td>
<td>14</td>
</tr>
<tr>
<td>A.3 Example Three: Multiple PABs</td>
<td>16</td>
</tr>
<tr>
<td>APPENDIX B LINK ERRORS &amp; MESSAGES</td>
<td>18</td>
</tr>
</tbody>
</table>
CHAPTER 1

INTRODUCTION

The OASIS Linkage Editor is a command program that is used to "link" together the output of an assembly or compilation process into an executable load module. This is a necessary step that follows the source program assembly or compilation of any problem program (except those programs written for the EXEC language processor, the OASIS BASIC compiler/interpreter, or the OASIS COBOL compiler).

Every program is designed to fulfill a particular purpose. To achieve that purpose, the program can generally be divided into logical units that perform specific functions. A logical unit of coding that performs a function, or several related functions, is a module. Ordinarily, separate functions should be programmed into separate modules, a process called modular programming.

Each module is separately assembled or compiled by one of the language translators. The output from a language translator is a source module; the output from a language translator is an object module. Before an object module can be executed, it must be processed by the Linkage Editor. The output of the Linkage Editor is a load module.

```
Source Program
         |   /
Language Translator   ---------> Program Listing
         |   /
Object Program
         |   /
Linkage Editor   --------------> Map Listing
         |   /
Load Program
```

Any module is composed of one or more program address blocks (PABS). A PAB is a unit of coding (instructions and data) that is, in itself, an entity. A PAB is the smallest separately relocatable unit of a program.

Each module in the input to the Linkage Editor may contain symbolic references to PABS in other modules; such references are called external references. The symbol referred to by an external reference must be either the name of a PAB or the name of an entry point in a PAB. PAB names and entry names are called external names. By matching an external reference with an external name, the Linkage Editor resolves references between modules.

The following chapters discuss the syntax of the LINK command, the use and requirements of the input files to the Linkage Editor, and the output of the Linkage Editor.
The OASIS Linkage Editor is invoked by executing the LINK command. The syntax of the command is:

```
LINK [<fn> [<ft> [<fd>]]] [[<option>...[]]]
```

Where:

- `<fn>` Indicates the file name of the object file to be linked or the file name of the specifications file (see FILE option).
- `<ft>` Indicates the file type of the object file to be linked or the file type of the specifications file (see FILE option). A default file type of OBJECT will be used when option FILE is not specified and a default file type of LINK will be used when option FILE is specified.
- `<fd>` Indicates the file disk of the object file to be linked or the file disk of the specifications file (see FILE option). When no `<fd>` is specified the default file search sequence will be used (see OASIS System Reference Manual).

**LINK Options**

- **BOOT** Indicates that the output is a bootstrap loader. When this option is specified the first 256 bytes of the output of the linkage is written to sector 0 of the output drive.
- **DISK=x** Indicates the drive that the output listing file is to be written to. When this option is specified the linkage parameters and map are written to a disk file with a file name equal to the `<fn>`, a file type of LINKMAP and the drive specified by this option.
- **DRIVE=x** Indicates the drive that the output file is to reside on. When this option is not specified the output file will be on the same drive as the input file.
- **FILE** Indicates that the file description specified is the file description of the file containing the LINK control parameters.
- **MAP** Indicates that the linkage map is to be generated and output to the listing device. This is a default option.
- **NOMAP** Indicates that the linkage map is not to be generated.
- **NOPRINT** Indicates that the linkage parameters and map are not to be output to the printer. This is a default option.
- **NOTYPE** Indicates that the linkage parameters and map are not to be displayed on the console.
- **NOXREF** This option suppresses the cross reference table generation. This is a default option.
- **PRINTER[n]** Indicates that the linkage parameters and map are to be output to the printer specified. If n is not specified then PRINTER1 is used.
- **SYSTEM** Indicates that the output is a system file. For example: LINK CLASS2 (SYSTEM outputs the file SYSTEM.CLASS2:S).
- **TYPE** Indicates that the linkage parameters and map are to be displayed on the console. This is a default option.
- **USR** Indicates that the output is a BASICUSR file. For example: LINK UPPER (USR outputs the file UPPER.BASICUSR).
- **WORK=x** Indicates the drive to be used for the linkage work files. When this option is not specified the system disk will be used.
- **XREF** This option is not implemented as of this release.

When the LINK command is invoked with no file description specified the program will expect the specifications file to be entered from the console. In this mode the LINK prompt character (#) will be displayed when the LINK command is waiting for a command.
3.1 Specification File

The link specification file is the input file that controls the basic operations of the linkage process. This file may be a console file or a disk file. It is not necessary to use the specification file; in fact, the normal simple linkages don't use this file.

The specification file is normally used when two or more object modules are being linked together or when some parameters of the resulting load module need to be modified from the object code.

To use the console as a specification file do not specify a file description when invoking the Linkage Editor. For example: \texttt{>LINK (PRINT.}

To use a disk file as a specification file you must use the option FILE when invoking the Linkage Editor and there must be a file description specified. For example: \texttt{>LINK TEST (FILE PRINT SYSTEM NOMAP.}

In the following subsections the term \texttt{<expression>} refers to an arithmetic expression involving constants, previously defined symbols, and the operators $+$, $-$, $\ast$, and $/$. For example:

\begin{verbatim}
LABEL+23
LOC1+4*10H 1000H
\end{verbatim}

An expression is evaluated in a left to right manner with no operator precedence. Numeric constants may be in decimal or hexadecimal (trailing H). String constants are specified with single quote characters surrounding them.

3.1.1 DEFINE Command

The DEFINE command allows you to assign a value to a symbol. The format of the command is:

\begin{verbatim}
DEFINE <symbol>=<expression>
\end{verbatim}

Where:

\texttt{<symbol>} Specifies the symbol that is to be assigned a value. This symbol must have already been defined by one of the included object modules.

\texttt{<expression>} Specifies the value that is to be assigned to \texttt{<symbol>}.

The DEFINE command is normally used to resolve an unresolved reference.

3.1.2 END Command

The END command marks the end of the input specification file records and instructs the Linkage Editor to output the load module and the load map. The format of the command is:

\begin{verbatim}
END
\end{verbatim}

When the END command is encountered the Linkage Editor attempts to resolve any unresolved references by searching all attached disks for object files with a file name the same as an unresolved reference. When a qualifying object file is found an INCLUDE is performed on that file. This process is repeated until the end of the table of unresolved references is reached. (Note: including a file in this manner may cause more unresolved references to be formed.)

When all references have been resolved that can be resolved in this manner and there still remains one or more unresolved references an implied \texttt{LIST} command is performed and the Linkage Editor returns to the console for further commands.

If there are no more unresolved references the load module is created on the specified or default disk, the load map is output to the list device and the Linkage Editor is exited.

3.1.3 ENTRY Command

The ENTRY command allows you to specify the execution entry point of the load module. The format of the command is:

\begin{verbatim}
ENTRY <pointer>
\end{verbatim}
ENTRY <expression>

Where:

<expression> Indicates the address of the entry point.

An ENTRY command has precedence over any end-of-file instructions that might specify an execution entry point.

3.1.4 INCLUDE Command

The INCLUDE command is the primary command of the input specification file. The INCLUDE command instructs the Linkage Editor to locate, analyze and assimilate an object file into the load module. The format of the command is:

INCLUDE <module name>[,<module name>]

Where:

<module name> Indicates the name of the object file to be included at this time. The file type of the object file must be OBJECT. More than one module name may be specified with one INCLUDE command by separating the module names with commas.

When the Linkage Editor receives an INCLUDE command it searches the attached disks for the module and includes the text and instructions of that module into the load module being built.

3.1.5 IGNORE Command

The IGNORE command allows you to create a load module that contains unresolved references by instructing the Linkage Editor to ignore certain symbols. The format of the command is:

IGNORE <symbol>[,<symbol>]

Where:

<symbol> Indicates the symbol that is to be ignored by the Linkage Editor. More than one symbol may be specified with one IGNORE command by separating the symbols with commas.

When a symbol is ignored by the Linkage Editor in this manner it is important to note that the reference to it is not actually taken out of the text of the load module—it merely references relative address zero. You should not ignore a symbol whose reference code is actually executed—the results will be undefined.

3.1.6 LIST Command

The LIST command allows you to see all of the currently unresolved references. The format of the command is:

LIST

When the LIST command is encountered the Linkage Editor displays all currently unresolved references on the list device.

3.1.7 NAME Command

The NAME command allows you to specify a program name for the load module that is different from the default. (The default name will be the name of the first included object module.) The format of the command is:

NAME <fn>[,<ft>][,<fd>]

Where:

<fn> Indicates the file or program name of the load module.

<ft> Indicates the file type of the load module. When this parameter is not specified the default file type will be used. (The default file type is dependent upon options used in the LINK command.)

<fd> Indicates the file disk of the load module. When this parameter is not specified the default file disk will be used. (The default file disk is dependent upon options used in the LINK command.)
3.1.8 ORIGIN Command

The ORIGIN command allows you to change a relocatable load module into an absolute load module. The format of the command is:

\[
\text{ORIGIN } \langle\text{expression}\rangle
\]

Where:

\(\langle\text{expression}\rangle\) Specifies the address that the load module is to be loaded at.

The ORIGIN command causes the relocation table to be used to change all relocatable references to absolute references and changes the load module into an absolute command module (the relocation table is not included in the load module).

3.1.9 OVERLAY Command

The OVERLAY command is not implemented in this version of LINK.

3.1.10 QUIT Command

The QUIT command allows you to abort the linkage process without creating a load module. The format of the command is:

\[
\text{QUIT}
\]

The QUIT command might be used when it is discovered that there are object modules required that have not been assembled yet or when the linkage is merely a test to determine unresolved references.

3.1.11 REPLACE Command

The REPLACE command allows you to change references from one, possibly undefined, symbol to another symbol. The format of the command is:

\[
\text{REPLACE } \langle\text{symbol1}\rangle=\langle\text{symbol2}\rangle
\]

Where:

\(\langle\text{symbol1}\rangle\) Indicates the symbol that is to be replaced.

\(\langle\text{symbol2}\rangle\) Indicates the symbol that is to replace \(\langle\text{symbol1}\rangle\).

The REPLACE command provides an easy means of linking an unfinished program. For example, the program might have calls to subroutines that are unwritten as yet. The REPLACE command could be used to change these references to a dummy subroutine that does exist without making a lot of changes to the source program just for test purposes.

Please note that symbols, as used by the Linkage Editor, are symbols defined as entry points, not just labels used in the assembly process.

3.1.12 SET Command

The SET command allows you to change the values in the load module text. The format of the command is:

\[
\text{SET } \langle\text{expression}\rangle=\langle\text{data}\rangle[,,\langle\text{data}\rangle]...
\]

Where:

\(\langle\text{expression}\rangle\) Specifies the address, relative to the base address of the load module of the text to be changed.

\(\langle\text{data}\rangle\) Is a list, separated by commas, of values that the text is to be changed to.

The SET command is normally used in, and is invaluable for, the modifications of parameters, defaults, etc., of a program without the modification of the source program.

3.1.13 Comments

Comments may be placed in the specifications file by using the semicolon (;) character. The Linkage Editor treats all characters in a record following the semicolon as a comment and will merely include them in any listing file that it may
3.2 Object File

An OASIS object file is the primary output file from the MACRO assembler and the primary input file to the Linkage Editor. An object file is a binary stream, sequential format file of control and text records. Each record in an object file consists of a header section and a text section. The header section for each record contains three values:

\[<\text{record length}> <\text{record type}> <\text{PAB number}>\]

Where:

- **<record length>** Specifies the number of bytes in the record, including the record length byte.
- **<record type>** Specifies the type of record with one of the following codes:
  - 01 PAB definition record (P)
  - 03 Text record (T)
  - 05 Relocation record (R)
  - 07 Entry definition record (E)
  - 09 External reference record (X)
  - 08 PAB to PAB reference record (F)
  - 0F End of file record (Z)
- **<PAB number>** Indicates which PAB the data following refers to.

Following the header section of a record is the text section. This text section varies in structure from one record type to another. The following sub-sections describe the format of each record type. The letter in parentheses is the letter displayed by the OASIS LIST command for that record type.

3.2.1 PAB Definition Record (P)

The PAB definition record specifies the PAB name, type, base address, and length. The PAB number and length are relative to the current object file only.

\[<\text{header}> <\text{pab length}> <\text{pab name}> <\text{pab type}> <\text{base address}>\]

PAB types are coded as a number:

- 01 = absolute
- 02 = relocatable
- 04 = common relocatable

3.2.2 Text Record (T)

The text record is normally the most common type of record in an object file. It contains the assembled instructions and data constants as specified in the source program.

\[<\text{header}> <\text{start addr}> <\text{data}>[<\text{data}>]...\]

3.2.3 Relocation Record (R)

The relocation record specifies a list of addresses within a PAB that must have the load address of the PAB added to them to form accurate address references. Although relocation record(s) may appear anywhere in the object file before the end of file record it is normal for this type of record to immediately follow the text records affected (see examples).

\[<\text{header}> <\text{addr}>[<\text{addr}>]...\]

3.2.4 Entry Definition Record (E)

The entry definition record specifies an address within a PAB that has been specified as an entry point with the ENTRY instruction in the source program. Along with the address the entry label is specified.

\[<\text{header}> <\text{entry addr}> <\text{entry name}>\]

When the Linkage Editor encounters an entry definition record it saves the entry point location and label and also searches its unresolved references table looking for any references that can be resolved by this definition.
3.2.5 External Reference Record (X)

The external reference record specifies an address within a PAB that is a reference to a label specified as an externally defined label with an EXTRN instruction in the source program. The address and label referenced is specified in this record.

The Linkage Editor tries to resolve this external reference by matching it with its currently defined entry point locations (defined with the entry definition record). If no match is found the external reference data is saved in the unresolved references table.

3.2.6 PAB to PAB Reference Record (F)

The PAB to PAB reference record is a special type of external reference, similar to the external reference record. The main difference is that the referenced label was resolved by the assembler because the label was defined in another PAB of the same assembly.

The PAB to PAB reference record specifies a list of addresses within a PAB that are references to other PABs.

The Linkage Editor uses the information in this record by adding the referenced PAB's base address to the referencing address and also adds the referencing address to the relocation table.

3.2.7 End of File Record (Z)

This record indicates the logical end of file for the object program. Normally this record is only two bytes in length (count and record type). When the source program contained an END statement that specified a starting address this record will also note the starting address and its PAB number. In this latter case the record length will be five.

3.2.8 Object File Record Examples

The following example object records are displayed as the LIST command displays them. This differs from the actual contents of the records in that the record type is displayed with the letter code instead of the numeric code and addresses are displayed in normalized mode instead of Z80 mode (low byte first).

```
10 P 00 04A2 MAP  02 0000  PAB definition, record length 16, relative PAB number 0, PAB length of 1186, PAB name is MAP, PAB type is relocatable, base address of 0.
08 T 00 00D9 7E19FF  Text record, record length 8, relative PAB number 0, addresses starting at 00D9, text is 7E 19 FF at locations 00D9, 00DA, and 00DB respectively.
05 R 01 000A 0012 003F  Relocation record, record length 5, relative PAB number 1, relative addresses 000A, 0012, and 003F in that PAB must have the load address added to them before execution.
0D E 00 0039 MAP  Entry definition record, record length 13, relative PAB number 0, address 0039 is the entry point named MAP (trailing spaces added).
0D X 00 0052 HELPMSG  External reference record, record length 13, relative PAB 0, address 0052 references external label HELPMSG (trailing space added).
09 F 00 004F 02 0086 01  PAB to PAB reference record, record length 9, relative PAB number 0, address 004F is a reference to relative PAB number 2, address 0086 is a reference to relative PAB number 1.
05 Z 00 0039  End of file record, record length 5, execution address is 0039 in PAB 0.
```
The output of the Linkage Editor generally includes two files: the load module and the listing file. The load module may be one of two forms dependant upon whether the load module is absolute or relocatable.

Load modules, when output, are always output to a disk file. The listing file, when output, may be output to a disk file, the console (default), or to one of the attached printer devices, dependant upon the options specified to the Linkage Editor.

4.1 Absolute Load Module File

An absolute load module output by the Linkage Editor is an exact image of the program to be executed. The directory entry for this type of a file specifies the load address and the load address is the execution address. An example of this type of a load module is the SYSTEM.NUCLEUS:S.

4.2 Relocatable Load Module File

A relocatable load module output by the Linkage Editor consists of two sections. The first section is an exact image of the program to be executed if it were loaded at address zero. The second section is the relocation table for the first section.

This relocation table consists of variable length records with the first byte specifying the word count. Following the word count byte are two byte entries (words) that specify relative addresses of the load module that need to have the relocation constant added to them before program execution begins. The relocation constant is the load address of the program. This load address is not known until the program is actually loaded into memory by the system.

The directory entry of a relocatable load module contains a record count that includes the recount count of both sections. Records in the first section are always 256 bytes in length.

Most OASIS commands are distributed as relocatable load module files.

4.3 Map Listing File

The map listing file may be output to a disk file, the console or one of the printer devices, dependant upon the options specified to the LINK command. The DISK option will cause the listing file to be output to disk; the PRINTER option will cause the listing file to be output to one of the printers; the TYPE option will cause the listing file to be output to the console (default); the NOTYPE option will cause the listing file to be suppressed.

The map listing file consists of two sections. The first section is a listing of the input specifications file, including any comments.

The second section is a memory map of the load module created by the Linkage Editor. This map is a listing of the PABs used in the load module. Listed with each PAB is the memory region used by the PAB, the PAB type, and the entry points defined in the PAB in ascending address sequence.

At the end of the memory map the relative entry address is listed along with the total length of the load module.
(This page intentionally left blank)
APPENDIX A

LINK EXAMPLES

A.1 Example One: Simple, Single PAB

MACRO CLASS6

SYSTEM.CLASS6 - Hazeltine 1400-1500

Addr Obj-Code Line Source Statement

... (source code listing continues) ...

- 10 -
APPENDIX A: LINK EXAMPLES

70+; input char translate routine

71+

0036 72+TRANIN:

0037 IF .NUL. 73+ .NUL.
0038 ENDIF
0039 IF .NUL. 74+ .NUL.
0040 ENDIF
0041 IF .NUL. 75+ .NUL.
0042 ENDIF
0043 IF .NUL. 76+ .NUL.
0044 ENDIF

0045 IF .NUL. 77+ .NUL.
0046 ENDIF
0047 IF .NUL. 78+ .NUL.
0048 ENDIF
0049 IF .NUL. 79+ .NUL.
0050 ENDIF
0051 IF .NUL. 80+ .NUL.
0052 ENDIF

0053 IF .NUL. 81+ .NUL.
0054 ENDIF
0055 IF .NUL. 82+ .NUL.
0056 ENDIF
0057 IF .NUL. 83+ .NUL.
0058 ENDIF
0059 IF .NUL. 84+ .NUL.
0060 ENDIF

0061 OR A 85+ clear cy
0062 86+ defin cy
0063 87+ defin cy
0064 88+ defin cy
0065 89+ defin cy
0066 90+ defin cy
0067 91+ defin cy
0068 92+ defin cy
0069 93+ defin cy
0070 94+ defin cy
0071 95+ defin cy
0072 96+ defin cy
0073 97+ defin cy
0074 98+ defin cy
0075 99+ defin cy
0076 100+ defin cy
0077 101+ defin cy
0078 102+ defin cy
0079 103+ defin cy
0080 104+ defin cy
0081 105+ defin cy
0082 106+ defin cy
0083 107+ defin cy
0084 108+ defin cy
0085 109+ defin cy
0086 110+ defin cy
0087 111+ defin cy
0088 112+ defin cy
0089 113+ defin cy
0090 114+ defin cy
0091 115+ defin cy
0092 116+ defin cy
0093 117+ defin cy
0094 118+ defin cy
0095 119+ defin cy
0096 120+ defin cy
0097 121+ defin cy
0098 122+ defin cy
0099 123+ defin cy
0100 124+ defin cy
0101 125+ defin cy
0102 126+ defin cy
0103 127+ defin cy
0104 128+ defin cy
0105 129+ defin cy
0106 130+ defin cy
0107 131+ defin cy
0108 132+ defin cy
0109 133+ defin cy
0110 134+ defin cy
0111 135+ defin cy
0112 136+ defin cy
0113 137+ defin cy
0114 138+ defin cy
0115 139+ defin cy
0116 140+ defin cy
0117 141+ defin cy
0118 142+ defin cy
0119 143+ defin cy
0120 144+ defin cy
0121 145+ defin cy
0122 146+ defin cy
0123 147+ defin cy
0124 148+ defin cy
0125 149+ defin cy
0126 150+ defin cy
0127 151+ defin cy
0128 152+ defin cy
0129 153+ defin cy
0130 154+ defin cy
0131 155+ defin cy
0132 156+ defin cy
0133 157+ defin cy
0134 158+ defin cy
0135 159+ defin cy
0136 160+ defin cy
0137 161+ defin cy
0138 162+ defin cy
0139 163+ defin cy
0140 164+ defin cy
0141 165+ defin cy
0142 166+ defin cy
0143 167+ defin cy
0144 168+ defin cy
0145 169+ defin cy
0146 170+ defin cy
0147 171+ defin cy
0148 172+ defin cy
0149 173+ defin cy
0150 174+ defin cy
0151 175+ defin cy
0152 176+ defin cy
0153 177+ defin cy
0154 178+ defin cy
0155 179+ defin cy
0156 180+ defin cy
0157 181+ defin cy
0158 182+ defin cy
0159 183+ defin cy
0160 184+ defin cy
0161 185+ defin cy
0162 186+ defin cy
0163 187+ defin cy
0164 188+ defin cy
0165 189+ defin cy
0166 190+ defin cy
0167 191+ defin cy
0168 192+ defin cy
0169 193+ defin cy
0170 194+ defin cy
0171 195+ defin cy
0172 196+ defin cy
0173 197+ defin cy
0174 198+ defin cy
0175 199+ defin cy
0200 200+ defin cy

009F 201 DCA 6
009G 202+ DCA:

009H 203+; Hazeltine 1500

009I 204+; Hazeltine 1500

009J 205+ DEFINE HOME, 'DC2

009K 206+ DEFINE CLEAR, 'FS, 8CH

009L 207+ DEFINE EOC, 'CAN, 8CH

009M 208+ DEFINE EOL, 'ST, 8CH

009N 209+ DEFINE LEFT, BS

LINK Rev B
No assembly errors.

>LIST CLASS6.OBJECT:A (OBJECT

<table>
<thead>
<tr>
<th>P</th>
<th>O00</th>
<th>CODE</th>
<th>02</th>
<th>0000</th>
</tr>
</thead>
<tbody>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>80</td>
<td>T</td>
<td>0</td>
<td>0002</td>
<td>00FE203804CF40AF99FE10CA9F00215F00875F1600195E23567AB337C8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1A13FEBC28074F3CC8CF4018E3F7B72B6EDCF4C18987FDCEB057EC8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FDCB0572013FDE5CF30FD8E40FE1282B7C9FDCB05F673C9FDCB05B6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C6AFCF4EB7C037C900000000000000000000000000000000000000000000000000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0000000000000000000000000000000000000000000000000000000000000000</td>
</tr>
<tr>
<td>3D</td>
<td>T</td>
<td>0</td>
<td>007D</td>
<td>0000000000000000000000000000000000000000000000000000000000000000</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0061</td>
<td>B500</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>006D</td>
<td>006E</td>
</tr>
<tr>
<td>09</td>
<td>T</td>
<td>0</td>
<td>005B</td>
<td>C600</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00B0</td>
<td>7E12FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0077</td>
<td>B800</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>008F</td>
<td>BCO0</td>
</tr>
<tr>
<td>09</td>
<td>T</td>
<td>0</td>
<td>0086</td>
<td>7E18CFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>008D</td>
<td>C000</td>
</tr>
<tr>
<td>09</td>
<td>T</td>
<td>0</td>
<td>00C0</td>
<td>7E0F8CFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>006F</td>
<td>C400</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00C4</td>
<td>08FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>006A</td>
<td>C600</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00C5</td>
<td>10FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0093</td>
<td>C800</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00C8</td>
<td>7E0CFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0073</td>
<td>CB00</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00CB</td>
<td>7E0BFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00B1</td>
<td>CE00</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00CE</td>
<td>7E1AFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0085</td>
<td>D100</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>00D1</td>
<td>FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00B2</td>
<td>D200</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00BD</td>
<td>7E13FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0087</td>
<td>D500</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>00D5</td>
<td>FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0097</td>
<td>D600</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00D6</td>
<td>7E11FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0067</td>
<td>D900</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00D9</td>
<td>7E19FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0069</td>
<td>DC00</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00DC</td>
<td>7E1FFFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0063</td>
<td>DF00</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>00DF</td>
<td>FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0065</td>
<td>E000</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>00E0</td>
<td>FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0099</td>
<td>E100</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00E1</td>
<td>7E01FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>009B</td>
<td>E400</td>
</tr>
<tr>
<td>08</td>
<td>T</td>
<td>0</td>
<td>00E4</td>
<td>7E1FFFF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>0075</td>
<td>E700</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>0067</td>
<td>FF</td>
</tr>
<tr>
<td>07</td>
<td>T</td>
<td>0</td>
<td>00BB</td>
<td>E800</td>
</tr>
<tr>
<td>06</td>
<td>T</td>
<td>0</td>
<td>00E8</td>
<td>FF</td>
</tr>
</tbody>
</table>
APPENDIX A: LINK EXAMPLES

LINK version 5.4B
CLASS6
04/28/80 14:06 Page 1

Memory map for SYSTEM.CLASS6:S

<table>
<thead>
<tr>
<th>Code</th>
<th>Low</th>
<th>High</th>
<th>Length</th>
<th>Type</th>
<th>Entry Address</th>
<th>Addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>CODE</td>
<td>0000</td>
<td>00EA</td>
<td>00EB</td>
<td>REL</td>
<td>0000</td>
<td></td>
</tr>
</tbody>
</table>

Entry Address: 0000
Total Length: 00EB (235 decimal)

>DUMP SYSTEM CLASS6 S

Sector 1764 (Track=68, Sector=9)
0000: C33600FE 203F00CF 4EC9000F 10CA9F00
0010: 2150FD00 5F16001F 5E235C7A B337C81A
0020: 1F2A8FCC 074F3CCD FD4B8F01 FD7E07B7
0030: 28E8C4EC 185E7FBD CB057EC8 FDCB5576
0040: 2ACFDCDE DF3FDE12B 02B7C9FD
0050: CB05F637 CF9DCB05 B6CBAF8C 4E87D037
0060: CB05F637 00E000D9 00E0006F 00E000C4
0070: 00E000C4 00E000B6 00E000E9 00E00000
0080: 00E000D9 00E000B6 00E000E9 00E0006F
0090: 00E000C4 00E000B6 00E000E9 00E0006F
00A0: 7ECF400E 11CF407C C6604FCF 407DC620
00B0: 4FCF400E C97E12EE 7E188CFF
00C0: 7E188CFF EF010F08 7E17CE1F 01EB7F06
00D0: FF7E0101 FF7E0101 FF7E0101 FF7E0101
00E0: 00000000 00000000 00000000 00000000
00F0: 00000000 00000000 00000000 00000000

Sector 1765 (Track=68, Sector=10)
0100: 1A01000E 00110061 0077008F 008D0066
0110: 00E00099 00730089 00850083 00870097
0120: 00E00099 00730089 00850083 00870097
0130: 00B007D8 00B007D8 00B007D8 00B007D8
0140: 00B007D8 00B007D8 00B007D8 00B007D8
0150: 00B007D8 00B007D8 00B007D8 00B007D8
0160: 00B007D8 00B007D8 00B007D8 00B007D8
0170: 00B007D8 00B007D8 00B007D8 00B007D8
0180: 00B007D8 00B007D8 00B007D8 00B007D8
0190: 00B007D8 00B007D8 00B007D8 00B007D8
01A0: 00B007D8 00B007D8 00B007D8 00B007D8
01B0: 00B007D8 00B007D8 00B007D8 00B007D8
01C0: 00B007D8 00B007D8 00B007D8 00B007D8
01D0: 00B007D8 00B007D8 00B007D8 00B007D8
01E0: 00B007D8 00B007D8 00B007D8 00B007D8
01F0: 00B007D8 00B007D8 00B007D8 00B007D8
A.2 Example Two: Specification File

This example is a listing of the specification file used to link the OASIS NUCLEUS command. Note the abundant use of comments and the modularity of the object modules. This makes maintenance of the program easier and is the recommended practice for all programs other than the simple, single module code.

The various DEFINES, IGNOREs and SETs are used to customize various parameters to a specific configuration.

The file is named NUCLEUS.LINK and is used by entering the command:

>LINK NUCLEUS (FILE
NAME SYSTEM.NUCLEUS:A

; Start of user area

; Initial Program Loader

; Size all banks

; Interrupts, clocks

; Disk driver

; Bank select

; INT mode 1

; IPL init routine
SET NUCLEUS+3EH=255
SET NUCLEUS+46H=250
SET CLKSW=085H
SET LUB+0=0
SET LUB+8=16
SET LUB+9=16
SET UCB0+23=250,30,10
SET UCB1+23=250,30,10
SET UCB2+23=250,30,10
SET UCB3+23=250,30,10
SET UCB16+2=79
SET UCB16+3=23
SET UCB16+4=0
ORIGIN 0
END

; Multi-user switch
; 4 Mhz
; RTC avail, RST5=BP
; Disk = DEV1
; CONIN=SYSTEM.DEV1
; CONOUT=SYSTEM.DEV1
; Disk STP, SET

; Line length
; Page length
; Class = 6 (Hazeltine)
This example shows a simple program example that uses two PABs. This program is incomplete in that it only checks to see if the operator has requested a help message display. When the operator has not requested a help message the program exits. At this point is where the normal program logic would be coded, possibly using additional PABs or the same ones.

>EDIT EX3CODE ASSEMBLE
NEW FILE
EDIT
*INPUT
TITLE "Example 3 - multiple PABs"
ENTRY EX3CODE
EXTERN HELP,HELPMSG,PROG
EX3CODE:
  PUSH BC              ; Save drive code
  PUSH DE              ; Save sector
  PUSH HL              ; Save parameter pointer
  LD B,9               ; Length
  LD DE,HELP           ; Point to HELP lit
.TSTHELP:
  LD A,(DE)            ; Get byte
  CP (HL)              ; Compare with token
  JR NZ,.NOHELP        ; Branch if not equal
  INC DE               ; Else bump pointers
  INC HL               ;
  DJNZ .TSTHELP        ; Loop
  POP HL               ; Is HELP request - display
  POP DE               ; Restore regs
  POP BC               ;
  LD DE,HELPMSG        ; Point to help message
.PAGE:
  LD B,9               ; Point to CONOUT
  SC 59                ; Get lines/page
  LD B,C               ; Move to B reg
.LINE:
  LD A,(DE)            ; Get character
  OR A                 ; Test if end
  RET                  ; Return to OASIS if is
  SC 2                 ; Else display
  DJNZ .LINE           ; Loop
  SC 49                ; Wait at bottom of page
  JR .PAGE             ; Display next page
.NOHELP:
  POP HL               ; Restore regs
  POP DE               ;
  POP BC               ;
  JP PROG              ;
END EX3CODE

*FILE
"EX3CODE.ASSEMBLE:A" filed

>EDIT EX3HELP ASSEMBLE:A
NEW FILE
EDIT
*INPUT
TITLE 'Example 3 - Help Message Data'
ENTRY HELP,HELPMSG
HELP:
  DC '.HELP',13
HELPMSG:
  DC 'Function: To illustrate an example',13
  DC 'of a multi-PAB program',13
  DC 'linkage',13
  DC 13
  DC 'Syntax: EX3CODE [(options[])]',13
  DC 13
  DC 'Where options are:',13
  DC 'PRINTERn output to printer # n',13
  DC 'TYPE output to the console',13
  DC 'NOTYPE suppress output',13
  DC 0
END

*FILE
"EX3HELP.ASSEMBLE:A" filed

>EDIT EX3PROG ASSEMBLE A
NEW_FILE
EDIT
#INPUT
TITLE 'Example 3 - Program'
ENTRY PROG
EX3CODE: REL
PROG:
  XOR A ; Clear return code
  SC 0 ; Exit
  END

#FILE
"EX3PROG.ASSEMBLE:A" filed

>MACRO EX3CODE
Pass one
Pass two
No assembly errors

>MACRO EX3HELP
Pass one
Pass two
No assembly errors

>MACRO EX3PROG
Pass one
Pass two
No assembly errors

>LINK
#include EX3CODE,EX3PROG,EX3HELP
#NAME EX3:S
#END

LINK version 5.4B

Memory map for EX3.COMMAND:A

<table>
<thead>
<tr>
<th>PAB-name</th>
<th>Low</th>
<th>High</th>
<th>Length</th>
<th>Type</th>
<th>Entry</th>
<th>Addr</th>
</tr>
</thead>
<tbody>
<tr>
<td>EX3CODE</td>
<td>0000</td>
<td>002B</td>
<td>002C</td>
<td>REL</td>
<td>EX3CODE</td>
<td>0000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>PROG</td>
<td>002B</td>
</tr>
<tr>
<td>EX3HELP</td>
<td>002F</td>
<td>011B</td>
<td>00F0</td>
<td>REL</td>
<td>HELP</td>
<td>002C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>HELPMSG</td>
<td>0038</td>
</tr>
</tbody>
</table>

Entry Address: 0000
Total Length: 011F (287 decimal)
APPENDIX B

LINK ERRORS & MESSAGES

** File "xxxxxxx.OBJECT" not found

This message is displayed following an INCLUDE command of a file that cannot be found on any of the attached disk drives.

** Including

This message is displayed when the Linkage Editor is performing automatic includes following an END command.

** Invalid character in expression

This message is displayed when an invalid expression is detected. Expression may only contain valid symbols (one to eight characters in length, must start with a letter and contain only letters, digits, dollar signs, and periods) numeric constants (must start with a digit and contain only digits, the letters A through F, and may be terminated with the letter H) and the arithmetic operators: + - * /.

** Invalid command

** Not Implemented

** Relocation error

Indicates that an expression containing relocatable symbols is in error. Usually the error is one of the following: a difference between two relocatable symbols of different PABs; the sum of two relocatable symbols; the product of two relocatable symbols; the quotient of two relocatable symbols; the product or quotient of a relocatable symbol and an absolute symbol.

** Too many segments

Up to 128 segments or object modules may be included in one linkage.

** Undefined symbol

An expression using symbols or the DEFINE, IGNORE, or REPLACE command reference an undefined symbol (a symbol not specified by an entry or external definition record).