COMPUTER ONE LINKER

The linker file supplied on the Assembler microdrive cartridge allows the assembly language programmer to write his program in separate sections which can be assembled independently. These sections are then joined by the linker to make a single executable program.

The linker takes an assembled file and fills in the addresses of any labels in the file which refer to code in other files. These **external references** are indicated in an assembly language program by using the 'XREF' directive (see section 3.6 of the manual).

A label in an assembly language program which will be referred to by another code file must be marked using the 'XDEF' directive (see section 3.7 of the manual).

Example:

We want to write an assembly language program in 3 sections, which we will assemble separately. The first two parts call a subroutine (CURS_START) which is in the third section. To allow the linker to join these three files the first two must contain the command:

XREF CURS_START

before the subroutine is called and the third must contain the command:

XDEF CURS_START

before the label is used.

The linker is particularly useful for large assembly language programs. Splitting the program into a number of files means that changes may only require the re-assembly of a small source file, rather than one large file.

You can also create a library of useful routines which can be called by different programs and which do not need to be assembled each time they are required.

The linker is run as a QDOS job by using the EXEC command:

EXEC mdv1_linker

The linker is less than 4.5K and can be run at the same time as the editor and assembler. The user interface for the linker is similar to that of the assembler.

The first prompt is for the total code space required, which will be roughly the sum of the sizes of each assembled module to be linked. The next prompt is for a command file name from which the linker will take its input when it requires a file name or other user supplied data. If you just press ENTER, the input will be from the console.

All the data required by the following prompts is taken either from the command file specified, or from the console if just ENTER was pressed. Any command file lines whose first character is a semi-colon are ignored, thus allowing comments to be used in the file.

The linker prompts for each file to be linked. Each file is loaded before the next one is prompted for, thus allowing you to have link files on different devices. If you just press ENTER when prompted for a link file all the currently unresolved symbols are displayed and you are prompted for another link file. If you just press ENTER again the link finishes with no code file produced. If you just press ENTER at the link file prompt when all symbols have been resolved you will be asked for the linker options. The only option available is 'j', which is the same as the assembler j option and allows you to produce an EXECable QDOS file. The default is not to produce an EXECable file. **Note** that if a file error occurs when input is being taken from a command file, e.g. the specified link file not being found, the linker will abort the current link. If such an error occurs when input is from the console, you are reprompted for the file name.

Next you are prompted for the name of the code file. If you just press ENTER, the file produced will have the same name as the first link file specified, but with the extension '_cde' or '_exe'. When the output file has been produced you are asked for the data space required for the job, if the 'j' option was selected. If you just pess ENTER the default data size of 256 bytes is used. When the link has finished you are given the option of linking another file, killing the linker job, or pressing CTRL-C for another task.

If, when taking input from a command file, an error occurs at any of the prompts from the options prompt onwards, e.g. the options specified in the command file were invalid, the command file is closed and you will be prompted for the input from the console. This can be done because the actual linking has been completed.

EXAMPLE COMMAND FILE

This section gives an example of a command file. If you have unresolved symbols when using a command file, the unresolved symbols are displayed and the link will be stopped. The maximum line length is 40 characters, otherwise a buffer overflow error will occur.

```
; command file to link 3 files
; first, the three file names
; default extension is '_lnk'
; default drive is mdv2_
mdv1_file1
mdv1_file2
; next file is on mdv2_
file3
; now ENTER since no more link files
(ENTER)
; now we want the 'j' option
j
; now the code file name
; default extension will be '_exe'
myfile
; code file is mdv2_myfile_exe
; lastly, the data size
; let's make it 512 bytes
; last line MUST have ENTER at the end
512(ENTER)
```

ASSEMBLER UPDATE

The assembler now has an INCLUDE file facility. This allows the assember to take its input from files other than the source file during an assembly. When the assember reaches the end of the included file, it continues to read from the original source file. The syntax for INCLUDING files is:

$INCLUDE ‹filename›

The $INCLUDE must start at the first column of the line. Include files may not be nested, but the source file can have any number of INCLUDE files. If any assembler errors are found in the include file, they are all displayed in the source file under the line specifying the include file. If the given ‹filename› is not found the assembler aborts the current assembly. If the assembler listing facility is used with a source file which contains include files, the incrementing of line numbers is temporarily switched off, so that all lines in the include file will be displayed with the same number.

There is a file provided on microdrive called **conv_bas** which is a SuperBASIC program to allow you to alter the default device of the assembler, editor and linker. Instructions for use are given in the file.