

## File Format - Linking

[Top](#) [Previous](#) [Next](#)

"Linking" is a technique that was necessary under DOS where memory was limited. Under Windows, which gives programs access to large blocks of [memory](#), linking is seldom necessary.

Basically, "links" are a method of connecting the two files together. Links specify the stations in the old file that connect to the new file. For example, if the shot B22 to CD1 connects the old file to the new file, then B22 is the link and you would use it as a link. You can have up to 500 links between files. The value of linking is that after the compiler has been given all the linking stations, it can forget all other stations in the old file. This frees up a large block of memory.

Even though Windows generally gives you enough memory so that linking is unnecessary, there are still situations where linking is useful. First of all, since it frees memory, it could be used to combine several large caves into a huge cave system. Second, linked files compile slightly faster. Finally, with linking, you can combine two caves that have duplicate survey names. Normally, you would have to rename all conflicting stations; but with linking, the program "forgets" all the stations in the old file so there is no conflict.

Link stations should be placed in the make file after the filename. If there is more than one link station they should be separated by commas. Here is a simple example:

```
#OLDCAVE.DAT;
#NEWCAVE.DAT,B22,C17;
```

In this example, B22 and C17 are linking stations between OLDCAVE and NEWCAVE. You will notice that OLDCAVE has no links. This is because it is the first file to be processed, and it does not need to be connected to a previous file. You can combine links and [fixed stations](#) like this:

```
#TEST2.DAT,AB4,C1[M,1.2,2.3,3.4],C12;
```

If you are working with three or more files, you have to plan ahead. This is because you may have links between the first file and the third file. Since the program erases everything but the link stations between files, you must be sure to carry all the links from the first to the third file. Look at the following example:

```
FILE1      -      (No links)
FILE2      -      Needs: A22 (From FILE1)
FILE3      -      Needs: A16 (From FILE1), B14 (From FILE2)
```

FILE2 needs A22 as a link from FILE1. FILE3 needs two links, A16 from FILE1 and B14 from FILE2. Since FILE2 is processed before FILE3, and FILE3 needs A16 from FILE1, you must carry A16 into FILE2 even though FILE2 doesn't need it for its own processing. This is the way the Make file would look:

```
#FILE1.DAT;          /no links
#FILE2.DAT,A22,A16;
#FILE3.DAT,A16,B14;
```

The following Make file for Wind Cave illustrates a complex Make file. (Under Windows, this kind of complex Make file is no longer necessary unless you have duplicate station names.)

#WIND1.DAT;

#WIND2.DAT,

C41,F12,P9,C41,C40,UG30,NFP1,C39,SA'12,PP3, /from Wind1 to Wind2

JF65,JF109,L\*6,JF10, /from Wind1 to Wind3

KX37R,KY258R,JW1R,KY357, /from Wind1 to Wind5

CR1,KK32,SA9R,BX21,KK32; /from Wind1 to Wind4

#WIND3.DAT,

JF65,JF109,L\*6,JF10, /from Wind1 to Wind3

KX37R,KY258R,JW1R,KY357, /from Wind1 to Wind5

CR1,KK32,SA9R,BX21,KK32; /from Wind1 to Wind4

#WIND4.DAT,

KX37R,KY258R,JW1R,KY357, /from Wind1 to Wind5

CR1,KK32,SA9R,BX21,KK32, /from Wind1 to Wind4

MP74,MP28,PC2,KY349,KY326, /from Wind3 to Wind5

AA29,AA30,AA32,CR4,PC7,ZB1; /from Wind3 to Wind4

#WIND5.DAT,

SE202, /from Wind4 to Wind6

KX37R,KY258R,JW1R,KY357, /from Wind1 to Wind5

MP74,MP28,PC2,KY349,KY326, /from Wind3 to Wind5

KK20,K29,KK26,KF14,BB35,BB37,KQ45, /from Wind4 to Wind5

KA1,KO4,KI24,KK33,KK37,KK41,KK53,BB33,SD15;

#WIND6.DAT,

SE202; /from Wind4 to Wind6