## **Return from the BFS**

Since an MFS calls the next filing system using a CALL statement, the MFS remains resident while filing systems that follow are executing. As with any subroutine calls, however, when routines (other MFSs and the BFS) are finished with the call, control returns to the statement following the CALL @NEXT. FS( ... ) statement.

An MFS thus has two opportunities with each operation to manipulate the data: when the call and arguments first come down from BASIC+, and again when the arguments are being returned from the BFS. In effect, this gives an MFS a "pre-file-operation" and a "post-file-operation" capability.

For example, an MFS that encrypts and decrypts data would modify the data differently depending on whether a read or write operation were being performed. If the data were being written to the file, the MFS would encrypt the data before calling the next filing system routine in line. All subsequent MFSs, and the BFS, would be passed a record that was already encrypted.

However, when reading the data, the MFS would decrypt the record after it had returned from the BFS and from any MFSs that came after the encryption MFS. In this way, the encryption MFS could pass clear data back to the BASIC+ READ statement.

The fact that MFSs can be stacked in this way affects the design of an MFS. In some instances, the order of MFSs is critical, since the output from one MFS is the input for the next, or since one MFS might terminate a file operation before subsequent MFSs get an opportunity to examine the data.