Reading directories

(DSK, WDS, SCS, IDE, HDX)

With the newer mass storage devices around, like HFDC, SCSI and IDE, it is possible to create subdirectories on those devices. When writing a program to read and display the contents of a directory the user must actually enter the whole path of the directory to read. It is not good enough anymore to just ask for a drive letter as is done in the example program of the "Disk memory system" manual. It depends on the device service routine (DSR) of the peripheral if it is possible to create subdirectories on the storage device and if files and subdirectories are time-stamped. The following table summaries some of the possibilities of a few peripherals:

Peripherals	Devicename	Max # files	Max # directories	Time stamp
TI Diskcontroller	DSK1-DSK3	127 / disk	0	No
Myarc HFDC	DSK1-DSK4	127 / dir	3 / disk	Yes
	WDS1-WDS3	127 / dir	114 / dir	Yes
SCSI	SCS1-SCS7	127 / dir	114 / dir	No (TI)
				Yes (Mdos)
Th. Nouspikel IDE	IDE1-IDE4	127 / dir	114 / dir	Yes (V4)

Reading a directory is like reading a file but when opening a directory for reading the file characteristics (INPUT,RELATIVE,INTERNAL,FIXED) and record format is predefined. It is also a good practice not to define the record length, because when the record length is not defined, the peripherals DSR will (and should) fill this in. On some peripherals the record length is 38 bytes but when the peripheral is capable of time stamping the record length is 146.

In the example program below, a pathname must be entered (line 190) and the directory-file is opened (line 270). Device names and directory names must be separated with a '.' and the pathname must also end with a single '.' so the program default add this to the entered pathname. A directory name has a minimum length of 1 character and a maximum of 10 characters.

The example program also tries to guess how many records can be maximum expected (line 200-260). Unfortunately the EOF() statement does not work (except for IDE) when reading directories, but when using Extended basic this problem can be solved with an ON ERROR clause (see at the bottom of the example).

When the directory-file is opened, records can be read. The first record read is always the volume label and device information. The next records are file and directory information. A record always consists of a name (string) and 3 numbers or 15 numbers when the peripheral is capable if time stamping. Because the directory-file is read in INTERNAL format the record internally consist of a 1 byte string length, a name with a maximum of 10 characters and 3 or 15 times one byte with the size of a floating point number size (value=8) followed by 8 bytes of the floating point number, which makes a record size of 1+10+3*(1+8)=38 bytes or 1+10+15*(1+8)=146. The first three numbers have the following meaning:

1 - Record type:
    0 = Volume label
    1 = File type Display/Fixed
    2 = File type Display/Variable
    3 = File type Internal/Fixed

```
        4 = File type Internal/Variable
        5 = File type Program (binary)
        6 = Directory

        If a file type < 0 it means that the file is write protected.

    2 - If Record type is Volume label (0): Size of the device in 256 bytes
                                            sectors
        If Record type is File type  (1-5): Number of 256 bytes sectors the file
                                            occupies
        If Record type is Directory   (6) : Number of 256 bytes sectors the
                                            directory index occupies
                                           (2*sectors/alucation unit) but some
                                            peripherals just returns 0!

    3 - If Record type is Volume label (0): Number of free 256 bytes sectors
        If Record type is File type  (1-4): The maximum record length of the file
        If Record type is Program file (5): The value 0, but on some peripherals
                                            the size in bytes!
        If Record type is Directory    (6): The value 0

If the peripheral is capable of time stamping the next 12 numbers have the
following meaning for files and directories:

    Time of creation:
     4 - Seconds      (00-59)        (0 for Volume label)
     5 - Minutes      (00-59)        (0 for Volume label)
     6 - Hours        (00-23)        (0 for Volume label)
     7 - Day of month (01-31)        (0 for Volume label)
     8 - Month        (01-12)        (0 for Volume label)
     9 - Year         (00-99 or 0000-9999)  (0 for Volume label)

    Time of last change:
    10 - Seconds      (00-59)        (0 for Volume label and directories)
    11 - Minutes      (00-59)        (0 for Volume label and directories)
    12 - Hours        (00-23)        (0 for Volume label and directories)
    13 - Day of month (01-31)        (0 for Volume label and directories)
    14 - Month        (01-12)        (0 for Volume label and directories)
    15 - Year         (00-99 or 0000-9999)  (0 for Volume label and directories)

When all records are read and the end of the directory-file is not yet reached,
a filename or directory name with a length of 0 bytes is returned (line 350) but
if all directory-file records are read an I/O error will occur. (in the case of
the IDE peripheral there will be always a last record returned with an empty
name!)



Example


Simple catalog program:

    100 CALL CLEAR
    110 DIM TYPE$(6)
    120 TYPE$(0)="VOL "
    130 TYPE$(1)="D/F "
```

```
    140 TYPE$(2)="D/V "
    150 TYPE$(3)="I/F "
    160 TYPE$(4)="I/V "
    170 TYPE$(5)="PGM "
    180 TYPE$(6)="DIR "
    190 INPUT "PATH:":PATH$
    200 MAX=127
    210 DEV$=SEG$(PATH$,1,3)
    220 IF DEV$="WDS" THEN 260
    230 IF DEV$="SCS" THEN 260
    240 IF DEV$="IDE" THEN 260
    250 GOTO 270
    260 MAX=127+114
    270 OPEN #1:PATH$&".",INPUT,RELATIVE,INTERNAL,FIXED
    280 INPUT #1:NAME$,TYPE,SIZE,FREE
    290 PRINT "DISKNAME : ";NAME$
    300 PRINT "DISKSIZE :";SIZE
    310 PRINT "AVAILABLE:";FREE
    320 PRINT "USED     :";SIZE-FREE
    330 PRINT "........................."
    340 FOR IX=1 TO MAX
    350 INPUT #1:NAME$,TYPE,SIZE,RLEN
    360 IF LEN(NAME$)=0 THEN 450
    370 PRINT NAME$;TAB(12);SIZE;TAB(17);TYPE$(ABS(TYPE));
    380 IF RLEN=0 THEN 410
    390 IF ABS(TYPE)=6 THEN 430
    400 PRINT " ";STR$(RLEN);
    410 IF TYPE>0 THEN 430
    420 PRINT TAB(27);"P";
    430 PRINT
    440 NEXT IX
    450 CLOSE #1
    460 END

If using Extended basic
Change lines:
        340 ON ERROR 500
        440 GOTO 350

Remove lines:
    200 through 260

Add line:
    500 RETURN 460
```