FWILD$ - Find File with Wildcard Filename

The FWILD$ routine extends to the traditional LIST$ routine to allow a Wildcard Filename to be specified to filter out unwanted files from the list returned by LIST$.

1.   Invocation

To search a Global directory matching files with a wildcard specification code:

```
CALL FWILD$ USING fw table
```

where *fw* is a control block of the following format:

```
01    FW
  02   FWVERS      PIC 9(4) COMP         * Block version number
       VALUE 1                          * Must be 1
  02   FWMODE      PIC 9(2) COMP         * Wildcard mode
                                        * 0 = Find simple substring
                                        * > 0 = Reserved for future use
  02   FWOPTS      PIC 9(2) COMP         * Open/shared file options, bit flag
                                        * #00 = Parse all files
                                        * #01 = Ignore IN USE files
                                        * #02 = Ignore SHARED files
                                        * #04 = Ignore closed files
  02   FWSORT      PIC 9(2) COMP         * Sort flag
                                        * 0 = Don't sort returned files
                                        * 1 = Sort returned files ascending
                                        * 2 = Reserved for future use
  02   FWUNIT      PIC X(3)              * Unit to scan
  02   FWWILD      PIC X(8)              * Sub-string to search for
  02   FWLENG      PIC 9(2) COMP         * No of significant chars in FWWILD
  02   FWOFF1      PIC 9(2) COMP         * Offset of FWWILD in filename
                                        * 0 = Try all possible offsets
                                        * 1 - 8 = Try offset N
  02   FWOFF2      PIC 9(2) COMP         * Optional 2nd offset to test
                                        * 0 = Just use FWOFF1
                                        * 1 - 8 = Last offset to try
  02   FWEOS       PIC 9(2) COMP         * End of string flag
                                        * 0 = Substring followed by 0 chars
                                        * 1 = Substring followed by N chars
                                        * 2 = Substring followed by either
  02   FWENTS      PIC 9(4) COMP         * No of 8-char entries in table
  02   FWRETURN    PIC 9(4) COMP         * Actual no of filenames returned
  02   FWFOUND     PIC 9(4) COMP         * Actual no of filenames found
  02   FWTOTAL     PIC 9(4) COMP         * Total number of files checked
```

and *table* is an array of the following format:

```
01    TABLE
  02   T-ENTS      OCCURS n PIC X(8)     * Table of returned filenames
```

2.   STOP Codes and Exception Conditions

The following STOP codes may be generated by FWILD$:

STOP code	Description
12205	Invalid block version. FWVERS does not contain 1.
12206	Invalid wildcard mode. FWMODE does not contain 0.

12207	Zero-length table. FWENTS contains 0.
12208	Invalid sub-string length. FWLENG is outside the range 1 to 8.

The following EXIT codes may be returned by FWILD$:

EXIT code	$$COND	Description
12202	2	OPEN$ could not open the directory.
12205	5	FWOFF1 is too high.
12206	6	FWOFF2 is too high
12207	7	FWOFF2 is higher than FWOFF1
12208	8	Unexpected error from LIST$.

3.   Programming Notes

FWILD$ is only available with GSM SP-38, or later.

FWVERS must be set to 1.

FWMODE must be set to 0 to indicate a simple substring search with no specific wildcard characters. Other values for FWMODE are reserved for future use.

FWOPTS is a bit-flag that allows In-use files, Open Shared files, or even normal Closed files to be excluded from the search. The following bit values are supported:

    #01   If this bit is set ignore all files returned by LIST$ as IN USE
    #02   If this bit is set ignore all files returned by LIST$ as SHARED
    #04   If this bit is set ignore all files returned by LIST$ as normal

For example, an FWOPTS setting of 0 will apply the wildcard search on **all** files returned by LIST$, including IN USE and SHARED files; an FWOPTS setting of 3 will exclude any IN USE or SHARED files; an FWOPTS setting of 4 will only include IN USE or SHARED files.

The FWSORT flag allows the files returned in the file table to be sorted in either ascending (1) or descending (2) order.

The unit in FWUNIT is passed directly to OPEN$. Any exceptions returned by OPEN$ or LIST$ are reflected by FWILD$.

The sub-string to search for in each filename returned by LIST$ must be established in FWWILD. Currently, no wildcard characters are allowed in the sub-string. The length of the sub-string, between 1 and 8, must be established in FWLENG.

The FWOFF1 and FWOFF2 parameters allow the starting position of the target sub-string, within the filename, to be specified. Two offsets are available to search for a sub-string in different positions within the filename in a single call of FWILD$. The following combination of FWOFF1 and FWOFF2 are valid:

FWOFF1 value	FWOFF2 value	Meaning
0	Don't care	The target sub-string, in FWWILD, will be compared with each filename from character position 1 in the filename to character position MAX (see below).
1 to 8	0	The target sub-string, in FWWILD, will be compared with each filename from character position FWOFF1 only. Note that an exception will be returned if the value in FWOFF1 is higher than the MAX possible offset (see below).
1 to 8	1 to 8	The target sub-string, in FWWILD, will be compared with each filename from character position FWOFF1 in the filename to character position FWOFF2 in the filename. Note that an exception will be returned if the value in FWOFF1 or FWOFF2 is higher than the MAX possible offset (see below). An exception will also be returned if the value in FWOFF2 is higher than the value in FWOFF1. If the value in FWOFF2 is the same as FWOFF1 then only a single offset will be tested.

The value in FWOFF1 (and FWOFF2) must allow the sub-string to be contained wholly in the 8-character filename returned by LIST$. Thus, the maximum value allowed for either FWOFF1 or FWOFF2 depends on the FWLENG parameter:

$$MAX = 9 - FWLENG$$

Some examples should make this clear:

FWLENG value	FWOFF1 value	FWOFF2 value	Description
5	1	0	Search for the 5 character sub-string from the 1st character position of the filename only.
5	1	4	Search for the 5 character sub-string from the 1st, 2nd, 3rd and 4th character positions of the filename.
5	1	5	Invalid, FWOFF2 exceeds (9 – FWLENG)
1	0	Don't care	Search for the single character sub-string from the 1st, 2nd, 3rd, 4th, 5th, 6th, 7th and 8th character positions of the filename.
7	0	Don't care	Search for the 7 character sub-string from the 1st and 2nd character positions of the filename.
1	4	5	Search for the single character sub-string from the 4th and 5th character

			positions of the filename.

The FWEOS field determines how characters after the target sub-string within the filename are considered. The following values are supported:

0   The sub-string must be followed by no, or only SPACE, characters in the filename
1   The sub-string must be followed by one, or more, non SPACE characters in the filename
2   The sub-string can be followed by some or no characters in the filename

Some combinations of FWOFF1, FWOFF2 and FWEOS are illogical. For example, FWLENG=1, FWOFF1=8 and FWEOS=1 will return no matching filenames because the sub-string will be at the end of filename thus no non-SPACE characters can follow it.

Combinations of FWOFF1, FWOFF2 and FWEOS allow most of the wildcard formats that would normally be specified using explicit wildcard characters. For example, with "*" the wildcard indicator for any number of characters (including zero characters); and "&" the wildcard indicator for any number of characters (excluding zero characters) then:

Wildcard string	FWOFF1	FWOFF2	FWEOS
ABC*	1	0	2
ABC&	1	0	1
*ABC	0	0	0
&ABC	2	6	0
ABC	0	0	2
*ABC&	0	0	1
&ABC*	2	6	2
&ABC&	2	6	1

The total number of files in the directory considered by FWILD$ is returned in FWTOTAL. If FWOPTS is 0 then this will be the number of files in the directory. If FWOPTS is not 0 then some IN USE or shared files in the directory may be ignored so FWTOTAL may be less than the number of files in the directory.

The number of files that match the wildcard specification is returned in FWFOUND.

The number of files that match the wildcard specification and have been returned in the filename table is returned in FWRETURN. Note that FWRETURN will be between 0 and the number of available entries in the filename table specified by FWENTS.

If FWFOUND > (FWRETURN = FWENTS) then the filename table passed to FWILD$ was not large enough to contain all the matching filenames.

If FWSORT is 0 the files are returned in the order they are encountered in the directory. If FWSORT is 1 the filenames in the table will be sorted in ascending order. Other values of FWSORT are reserved for future use.

4.   Examples
[EXAMPLE REQUIRED]

5.   Copy-Books
None.

6.   See Also

OPEN$ Open Directory
LIST$ List Directory
CLOSE$ Close Directory