

# .PATH-problem [was Re: make or kmod.mk broken]

---

*Source:* <http://unix.derkeiler.com/Mailing-Lists/FreeBSD/current/2006-01/msg00402.html>

---

- *From:* Max Laier <[max@xxxxxxxxxxxxxxx](mailto:max@xxxxxxxxxxxxxxx)>
  - *Date:* Sat, 14 Jan 2006 21:03:32 +0100
- 

On Saturday 14 January 2006 18:08, Hartmut Brandt wrote:

> Max Laier wrote:  
>> On a related question: How can I get the actual location of a file that  
>> is in .PATH? All I could come up with was `${.ALLSRC:M*${MY_FILE}}` which  
>> doesn't work as I am explaining here.  
>  
> `M*${MY_FILE}` would also match 'foobar' if MY\_FILE is 'bar' which is  
> probably not what you want.  
> .IMPSRC might be what you want if you talk about an implicate rule.

What I am trying to do is parse a list of "filename:shortname"-objects. This is to support easy building of firmware modules. If things work as I want them to you can build a firmware module with a two line Makefile:

```
| KMOD=somefirmware  
| FIRMWS=firmfile1.fw:somename1 firmfile2.fw:somename2
```

and it works if the firmfiles are in the same directory, but if they are in .PATH it fails. I was looking at .IMPSRC initially as well, but failed to understand the concept :-\ ... any help greatly appreciated. Thanks.

My current hack is attached, the complete thing is in perforce at `//depot/user/mlaimer/firmware/...`

```
--  
/"\ Best regards, | mlaimer@xxxxxxxxxxxx  
\ Max Laier | ICQ #67774661  
X http://pf4freebsd.love2party.net/ | mlaimer@EFnet  
\ ASCII Ribbon Campaign | Against HTML Mail and News
```

```
Index: kmod.mk=====
RCS file: /usr/store/mlaimer/fcvs/src/sys/conf/kmod.mk,v
retrieving revision 1.200
diff -u -r1.200 kmod.mk
--- kmod.mk      29 Nov 2005 09:37:42 -0000      1.200
+++ kmod.mk      6 Jan 2006 03:48:58 -0000
@@ -36,6 +36,8 @@
#
# SRCS          List of source files.
#
```

## .PATH-problem [was Re: make or kmod.mk broken]

```
+# FIRMWS      List of firmware images in format filename:shortname
+#
# DESTDIR      The tree where the module gets installed. [not set]
#
# +++ targets +++
@@ -119,6 +121,24 @@
CFLAGS+=      -mlongcall -fno-omit-frame-pointer
.endif

+.if defined(FIRMWS)
+.if !exists(@)
+${KMOD:S/$/.c/}: @
+.else
+${KMOD:S/$/.c/}: @/tools/fw_stub.awk
+.endif
+
+   ${AWK} -f @/tools/fw_stub.awk ${FIRMWS} -m${KMOD} -c${KMOD:S/$/.c/g}
+
+SRCS+= ${KMOD:S/$/.c/}
+CLEANFILES+= ${KMOD:S/$/.c/}
+
+.for _firmw in ${FIRMWS}
+${_firmw:C/\:.*$/fwo/}:          ${_firmw:C/\:.*$//}
+   ${LD} -b binary ${LDFLAGS} -r -d -o ${.TARGET} ${_firmw:C/\:.*$//}
+OBJS+= ${_firmw:C/\:.*$/fwo/}
+.endfor
+.endif
+
+OBJS+= ${SRCS:N*.h:R:S/$/.o/g}

.if !defined(PROG)

#!/usr/bin/awk -f

#-
# Copyright (c) 2006 Max Laier.
# All rights reserved.
#
# Redistribution and use in source and binary forms, with or without
# modification, are permitted provided that the following conditions
# are met:
# 1. Redistributions of source code must retain the above copyright
#    notice, this list of conditions and the following disclaimer.
# 2. Redistributions in binary form must reproduce the above copyright
#    notice, this list of conditions and the following disclaimer in the
#    documentation and/or other materials provided with the distribution.
#
# THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS `AS IS' AND
# ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
# IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
# ARE DISCLAIMED.  IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
# FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
# DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
# OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
# HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
# LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
# OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
# SUCH DAMAGE.
#
# $FreeBSD$

#
# Script to generate module .c file from a list of firmware images
```

## .PATH-problem [was Re: make or kmod.mk broken]

```
#

function usage ()
{
    print "usage: fw_stub <firmware:name>* [-m modname] [-c outfile]";
    exit 1;
}

# These are just for convenience ...
function printc(s)
{
    if (opt_c)
        print s > ctmpfilename;
    else
        print s > "/dev/stdout";
}

BEGIN {

#
# Process the command line.
#

num_files = 0;

for (i = 1; i < ARGV; i++) {
    if (ARGV[i] ~ /^-/) {
        #
        # awk doesn't have getopt(), so we have to do it ourselves.
        # This is a bit clumsy, but it works.
        #
        for (j = 2; j <= length(ARGV[i]); j++) {
            o = substr(ARGV[i], j, 1);
            if (o == "c") {
                if (length(ARGV[i]) > j) {
                    opt_c = substr(ARGV[i], j + 1);
                    break;
                }
            } else {
                if (++i < ARGV)
                    opt_c = ARGV[i];
                else
                    usage();
            }
        }
    } else if (o == "m") {
        if (length(ARGV[i]) > j) {
            opt_m = substr(ARGV[i], j + 1);
            break;
        }
    } else {
        if (++i < ARGV)
            opt_m = ARGV[i];
        else
            usage();
    }
} else {
    if (k = index(ARGV[i], ":")) {
        filenames[num_files] = substr(ARGV[i], 1, k - 1);
        shortnames[num_files] = substr(ARGV[i], k + 1);
    }
}
}
}
```

## .PATH-problem [was Re: make or kmod.mk broken]

```
        } else {
            filenames[num_files] = ARGV[i];
            shortname[num_files] = ARGV[i];
        }
        num_files++;
    }
}

if (!num_files || !opt_m)
    usage();

cfilename = opt_c;
ctmpfilename = cfilename ".tmp";

printf("#include <sys/param.h>\n\
#include <sys/errno.h>\n\
#include <sys/kernel.h>\n\
#include <sys/module.h>\n\
#include <sys/linker.h>\n\
#include <sys/firmware.h>\n");

for (file_i = 0; file_i < num_files; file_i++) {
    symb = filenames[file_i];
    # '-', '.' and '/' are converted to '_' by ld/objcopy
    gsub(/-|\.|\/, "_", symb);
    printf("extern char _binary_" symb "_start[], _binary_" symb "_end[];");
}

printf("\nstatic int\n\
opt_m "_fw_modevent(module_t mod, int type, void *unused)\n\
{\n\
    struct firmware *fp;\n\
    switch (type) {\n\
        case MOD_LOAD:");

for (file_i = 0; file_i < num_files; file_i++) {
    short = shortnames[file_i];
    symb = filenames[file_i];
    # '-', '.' and '/' are converted to '_' by ld/objcopy
    gsub(/-|\.|\/, "_", symb);

    if (file_i == 0)
        reg = "\t\tfp = ";
    else
        reg = "\t\t(void)";

    reg = reg "firmware_register(\"" short "\", _binary_" symb "_start , ";
    reg = reg "(size_t)(_binary_" symb "_end - _binary_" symb "_start), ";

    if (file_i == 0)
        reg = reg "NULL);";
    else
        reg = reg "fp);";

    printf(reg);
}

printf("\t\treturn (0);\n\
        case MOD_UNLOAD:");

for (file_i = 1; file_i < num_files; file_i++) {
    printf("\t\tfirmware_unregister(\"" shortnames[file_i] "\");");
}
```

.PATH-problem [was Re: make or kmod.mk broken]

```
}

printf("\t\tfirmware_unregister(\" " shortnames[0] " \");");

printf("\t\treturn (0);\n
      }\n
      return (EINVAL);\n
}\n
\
static moduledata_t " opt_m "_fw_mod = {\
  \" " opt_m "_fw\", \
  " opt_m "_fw_modevent\", \
  0\
};\
DECLARE_MODULE(" opt_m "_fw, " opt_m "_fw_mod, SI_SUB_DRIVERS, SI_ORDER_FIRST);\
MODULE_VERSION(" opt_m "_fw, 1);\
MODULE_DEPEND(iwi_boot_fw, firmware, 1, 1, 1);\
");

if (opt_c)
    if ((rc = system("mv -f " ctmpfilename " " cfilename))) {
        print "'mv -f " ctmpfilename " " cfilename "' failed: " rc \
              > "/dev/stderr";
        exit 1;
    }

exit 0;

}
```

**Attachment:** [pgpIZhQtQpp1p.pgp](#)

*Description:* PGP signature