Using Floating Point Math with the (A)DNP/1486 Embedded Linux

**** 1. Step***: A very simple C source code with floating point math. Don't forget the include for the header file `math.h`.

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

int main (void)
{
   float x, y;

   printf ("\nHello Embedded Linux User !!!\n\n");
   x = 1.36;
   y = sin(x);
   x = asin(y);
   return (0);
}
```

**** 2. Step***: The sample makefile for building the executable file. This makefile works also with the GNU C cross compiler for the DIL/NetPC DNP/1110 with a StrongARM processor.

```
PROJ    =    hellomath
#host compiler
CROSS   =
CCFLAGS =

#arm cross compiler (DNP/1110-3V)
#CROSS  =    /usr/local/arm-linux/bin/arm-linux-
#CCFLAGS =   -march=armv4 -mtune=strongarm

#i486 cross compiler
#CROSS  =    /usr/local/i486-linux/bin/i486-linux-
#CCFLAGS =   -march=i386

CC =    $(CROSS)gcc
CFLAGS  =    -Wall -Os $(CCFLAGS)

#dynamic libc and dynamic libm (you need libm in system !)
#LFLAGS =    -Wl,-s -lm

#static libc and static libm (big file !)
```

```
#LFLAGS =    -Wl,-s -static -lm

#dynamic libc and static libm (a good choice !)
LFLAGS  =    -Wl,-s -Wl,-Bstatic -lm -Wl,-Bdynamic

$(PROJ): $(PROJ).c Makefile
   $(CC) $(CFLAGS) $(PROJ).c -o $(PROJ) $(LFLAGS)

clean:
   rm -f $(PROJ) $(PROJ).o
```