

Cfmake ColdFire Maker

rev. 1.56

Contents

| | |
|-------------------------------|----|
| 2. General..... | 5 |
| 2.1 Command Line Options..... | 7 |
| 2.2 Making..... | 9 |
| 2.3 Make File Format..... | 11 |
| 3. Errors..... | 13 |
| 4. Limitations..... | 15 |
| 4.1 Known Bugs..... | 15 |
| 5. Changes..... | 17 |

2. General

The ColdFire Maker , cfmake, is a freeware project maker for ColdFire processors.

The job of the maker is to compile, assemble and link all the files in a project. It does this through a makefile, which is a simple ascii document describing how the project should be made.

2.1 Command Line Options

Usage: `cfmake [options] makefilename`

Options:

| | |
|-------------------|---|
| <code>none</code> | Compile, Assemble & Link files. |
| <code>-s</code> | Assemble files. |
| <code>-c</code> | Compile files. |
| <code>-d</code> | Display compilation. (otherwise write to log files) |
| <code>-f</code> | Force compile. This will disregard file age. |
| <code>-i</code> | Ignore compilation errors. |
| <code>-k</code> | Keep compiler-assembled files. |
| <code>-l</code> | Link files. |

Note that if no options are given, `cfmake` assumes `'-c -a -l'`. If you only wanted to link files (without compile or assemble), then use `-l` for example. If the option `'-f'` is not given, `cfmake` will only compile files newer than the object file date.

2.2 Making

The usual format for making looks like this;

```
cfmake mymakefile
```

where:-

```
cfmake  
mymakefile
```

- is the name of the executable.

- is the name of your custom project makefile in ascii format. You may use the extensions “.m”, “.mak” and “.make” on your filename. Just typing the main filename will work since cfmake will append each extension and try loading the file. In this example, if the makefile was actually called “mymakefile.make”, then you only need to type “mymakefile”.

Cfmake is designed to be used with other ColdFire freeware products from Austex Software. These are cfcc (C Compiler), cfasm (Assembler) and cfink (Linker).

2.3 Make File Format

Inside each makefile are options on compilation, assembling and linking for a project. The project descriptions are divided into blocks. These are enclosed in braces '[' and ']'. Valid block names are:-

```
[Compile]
[Assemble]
[Link]
[LinkLibraries]
[Output]
[GlobalCompile]
[GlobalAssemble]
[GlobalLink]
```

Under each of these descriptions are separate lines of each file/option within that block. So, if you have three files that you want to compile, you would fill the makefile out as:-

```
; Mymakefile project #1234
; Copyright Widgets Pty Ltd 2003
[Compile]
    myfile1.c
    myfile2.c
    myfile3.c
[Assemble]
    myfile1.s
    myfile2.s
    myfile3.s
[Link]
    myfile1.o
    myfile2.o
    myfile3.o
[LinkLibraries]
    mylibrary.lib
```

Comments are denoted by a semicolon ';' and can be used anywhere within the file. The output file is setup by the output block:-

```
[Output]
    ; My rom image filename
    myfile.rom
```

If no output block is specified, the default filename generated is 'a.out'. Global options for compiling, assembling and linking are also defined within their own blocks:-

Make File Format

```
[GlobalCompile]
    ; Compiler options
    -x -y -z

[GlobalAssemble]
    ; We want object file output for each file
    -o

[GlobalLink]
    ; We want build information and the ROM address at $78000000
    -i -r78000000
```

When the make program is initiated, each file under the [Compile] block will be compiled with the options in [GlobalCompile]. The same operation applies for assembling and linking, so assembly uses [GlobalAssemble] and linking uses [GlobalLink].

For libraries, there is the [LinkLibraries] block.

3. Errors

Error messages are meant to be self-explanatory. The general error messages that you may encounter are:-

File <name> not found

One of the required files could not be found

<name> does not belong in a valid block

This usually occurs when a block name is incorrect, or you have not defined a block name at the beginning of a file.

Block type <name> is not valid

The <name> of the block is not valid. The names are not case sensitive, however they do need to be spelled correctly.

Could not compile file. Please check...

This could be caused by a number of things. Check that the compiler (cfcc) is installed correctly. Also, that the file you are compiling exists and that the [GlobalCompile] options are correct.

Could not assemble file. Please check...

This could be caused by a number of things. Check that the assembler (cfasm) is installed correctly. Also, that the file you are assembling exists and that the [GlobalAssemble] options are correct.

Could not link file. Please check...

This could be caused by a number of things. Check that the linker (cflink) is installed correctly. Also, that the files you are linking exist and that the [GlobalLink] options are correct.

4. Limitations

1. The maker does not support other compiler, assembler or linker utilities for use with ColdFire processors.

4.1 Known Bugs

There are no known program bugs for this current version.

If you find any program bugs, then please send reports to:-

bugs@austexsoftware.com

5. Changes

- 1.56 C Compiler error files of size zero (indicating no errors) are now removed.
- 1.54 Compiler error files now removed before a file is re-compiled. (less mess...)
- 1.52 Bumped for new release.
- 1.50 Added library handling.
- 1.40 File date comparison now works with filenames with more than a one character extension (e.g. .asm).
- 1.30 Added filename extension handling.
- 1.20 Added ppc version.
- 1.10 Added ignore compilation errors flag.
- 1.00 Modified error message. Program file deleted before linkage.
- 0.98 Fixed problem with cli file generation.

*Copyright (C) 2001-2003 Austex Software
All rights reserved.
www.austexsoftware.com*

