

# *FEAP - - A Finite Element Analysis Program*

---

*Version 8.2 Installation Manual*

Robert L. Taylor  
Department of Civil and Environmental Engineering  
University of California at Berkeley  
Berkeley, California 94720-1710  
E-Mail: rlt@ce.berkeley.edu

May 2008

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Installations</b>	<b>3</b>
2.1	UNIX/Linux and Apple Computer Installations . . . . .	3
2.1.1	Editing files <code>makefile</code> and <code>makefile.in</code> . . . . .	3
2.1.2	Installing the program . . . . .	5
2.1.3	Running <i>FEAP</i> . . . . .	5
2.2	Windows Installation: COMPAQ/Intel Visual Fortran . . . . .	6
2.2.1	Build of Library . . . . .	6
2.2.2	Build of Executable . . . . .	8
2.2.3	Alternate Windows graphics form . . . . .	9
2.2.4	Running <i>FEAP</i> . . . . .	9
2.3	Installation with no graphics . . . . .	10

# Chapter 1

## Introduction

The source files for the *FEAP* system are delivered on one CD-ROM. In addition, the disk contains printable files (using Acrobat Reader) for the manuals. The program is furnished under license by the Department of Civil and Environmental Engineering at the University of California, Berkeley. It is for use by the licensee only and may not be redistributed to others in any form without prior authorization by the University of California, Berkeley.

It is recommended that a directory with the name 'feap', or similar be created and all information on the CD-ROM copied into this directory. The source files will reside in a directory with the name 'ver82'. When the files are copied to the 'feap' directory the program structure will have the directory structure shown in Table 1.1.

Within the licensed unit it is permitted to make public versions of the following:

1. An executable version and/or an archive (library) file(s);
2. The files 'feap82.f' (main program); 'contact.f' (dummy file to eliminate contact module); 'pplotf.f' (dummy file to eliminate graphics).
3. include files; and
4. User files in the directory 'user' (e.g., 'elmt01.f', 'umacr1.f', 'umesh1.f', 'usetm1.f', etc.).

All other files are considered to be the property of the licensee and should not be made available to others.

The routine 'feap82.f' may be modified to set parameters as necessary. The current *FEAP* system uses dynamic memory allocation for all the main arrays during solution.

As such the maximum size of problems that can be solved by the program is limited only by the available memory of the computer used.

If the compiler used does not have routines for `malloc` and `free` the system may be installed using the routines in the directory ‘patch/memory’. This form uses an unlabeled common to store the arrays. The array `mr` in the routine subprogram `pinitm.f` can be resized by setting the parameter `mrmax` to an appropriate value for the computer used.

Please report any installation problems by e-mail to: [feap-help@vulture.ce.berkeley.edu](mailto:feap-help@vulture.ce.berkeley.edu).

```

feap
ver82
  contact -----+- main
  elements ----+   |- ntrnd
  main          |   |- nts2d
  maintain      |   |- nts3d
  plot          |   |- ptpnd
  program       |   |- tie2d
  unix          |   +- util
  user          +----- elements
  windows                          |
  packages ----+- arpack --+ archive |- frame
  unix          +- blas              |- material
  user          +- lapack             |   |- small
  window2      +- meshmod            |   +- finite
  include -----+- integer4        |- shells
  parfeap ----+      +- integer8     |- solid1d
  parfeap ----+- partition            |- solid2d
  +- unix              |- solid3d
  +- windows           |- thermal
  +- packages ----+
  patch-----+      +- arpack
  +- hp
  +- ibm
  +- memory

```

Table 1.1: Directory structure for *FEAP* system

# Chapter 2

## Installations

### 2.1 UNIX/Linux and Apple Computer Installations

The build in a UNIX/Linux environment is controlled by the data contained in files `makefile` and `makefile.in` located in the directory "ver82".

If your system has a revision control system (RCS) or (CVS) it is highly recommended that *FEAP* source files utilize this feature. This permits changes to the program without losing any previous versions. This is a basic decision which must be made before proceeding with additional installation steps.

#### 2.1.1 Editing files `makefile` and `makefile.in`

Use a text editor to make changes to the files `makefile` and `makefile.in` located in the directory `ver82`. The location of specific parts of the *FEAP* program is controlled by the parameter `$(FEAPHOME8_2)`. This parameter may be set using the system command:

```
setenv FEAPHOME8_2 /home/.../feap/ver82
```

or for *bash*

```
export="FEAPHOME8_2 /home/.../feap/ver82"
```

where the last parameter is the path name where the individual subdirectories in `ver82` are located. Alternatively, the command may be inserted within the `.cshrc`, `.tcshrc` or whatever user filename is located in the user root directory.

**Editing `makefile.in`**

It is necessary to edit the file `makefile.in` as indicated below. Note that comments in this file are set by placing the character `#` in the first column.

Edit file `makefile.in` as follows:

1. Select the appropriate include files to use for the `FINCLUDE` parameter. For 32-bit machines use the line with `integer4`; for 64-bit machines use the line with `integer8`. Some systems also require a path for the C-includes. The path is assigned to the `CINCLUDE` parameter.
2. In section *Which compilers to use* set the name of your Fortran compiler after `FF =` (Different options are indicated with all but one commented with the `#` symbol). Also set the name of your C compiler after `CC =` (Again, different options are indicated).
3. Set optimization level to use. Currently this is set to `O2` and the flag for all warnings also is active.
4. In Section *Source Types*:
  - (a) If you will use RCS, set `FSOURCE = RCS/` and `CSOURCE = RCS/` (You may set `# FSOURCE =` and `# CSOURCE =` as these are treated as comments).
  - (b) If you will not use RCS, set `FSOURCE =` and `CSOURCE =` (You may set `# FSOURCE = RCS/` and `# CSOURCE = RCS/` as these are treated as comments).
5. In the section *Source Extender*:
  - (a) If you will use RCS, set `FEXT = f,v` and `CEXT = c,v` (You may set the `#FEXT = f` and `#CEXT = f` lines as comments).
  - (b) If you will not use RCS, set `FEXT = f` and `CEXT = c` (You may set the `#FEXT = f,v` and `#CEXT = f,v` lines as comments).
6. Generally, no options are needed for `FOPTIONS =` or `COPTIONS =`; however, if you experience difficulties some may need to be inserted.
7. In section *What options to be used by the loader* select the correct X-library (i.e., either 32-bit (`lib`) or 64-bit (`lib64`)). If a non-standard installation is made some changes may be required.
8. In section *What archiving to use* standard options are given. Usually no change is necessary.

### Editing makefile

Generally, this file does not need any modifications unless new options are to be added.

## 2.1.2 Installing the program

If necessary, change directories until you are in `ver82`. Also, if the `RCS` option is used it is necessary to first issue the command

```
make checkout
```

This will check all the files out from `RCS` in *read only* mode – that is, they may not be edited and subsequently written back to the disk.

The installation of the program is made using the command:

```
make install
```

Each subdirectory should be processed and the compiled object files placed in the archive named in the `makefile.in`. A successful compilation should deposit the executable (named `feap`) in the subdirectory `main`.

If errors occur it is necessary to correct them and then recompile the program using the command `make install`.

## 2.1.3 Running *FEAP*

After a successful installation step the *FEAP* program is ready for use. To permit running the program from any directory it is convenient to define a path to the location of the executable. This may be done by placing the one of the following lines in an appropriate file in the root directory;

```
alias feap '/fullpath/ver82/main/feap'
```

or

```
alias feap="/fullpath/ver82/main/feap"
```

where `fullpath` is the complete path to the `ver82` directory.

The program may now be executed from any directory by first preparing an input file (see the *User Manual* for preparing this file) and issuing the instruction

```
feap
```

from the command line in any window. If graphics is to be enabled it may be necessary to create an X-window using the command:

```
startx &
```

Full testing requires the preparation of an *input file* as described in the *FEAP User Manual*.

## 2.2 Windows Installation: COMPAQ/Intel Visual Fortran

An executable version of *FEAP*, including all graphics options, may be built using the Compaq or Intel Visual Fortran compiler.

Generally, it is desirable to place all parts of the program except the main program (`feap82.f` in the `main` directory) into a single library and then finally build a main (executable) program. For example, a build with the library named *lib82* places all basic parts of the program together. A main program called *feap* may then be constructed which includes this library. However, alternate combinations separating the library into parts may be selected. build is described for the name given above.

### 2.2.1 Build of Library

The following steps may be used to build the necessary library for the *FEAP* program:

1. Open the Developer Studio for a new project.
2. Under *File* select *New*. (N.B. Options to be selected are shown in italics).
  - (a) Under *Projects* tab select *Fortran static library*. **Do not select a dynamic link library (DLL)**.
  - (b) In *location* window set path to a location for build files. The path must exist, if not use standard Windows steps to create the folder before doing this step.
  - (c) In *Project name* assign a library name (e.g., lib82). (N.B. Items to be selected and named by the user are indicated by underlines).

- (d) Press *OK* button to start (N.B. small upper window should now have the notation **Workspace program**).
3. Under *Build*:
    - (a) Select *Set Active Configuration* and choose between *Release* and *Debug* (generally I use *Release* for most builds, however, if you use *Debug* it will be necessary to set the compile option to ignore array bounds).
  4. Under *Project* select *Project Settings*:
    - (a) Choose *Fortran* tab and set Category window to *Preprocessor*.
    - (b) In *INCLUDE and USE paths* window insert the path to where the include files are located. (The path will generally be set when you install the program - e.g., `c:\feap\ver82\include` and `c:\feap\ver82\include\integer4` – or `integer8` for 64-bit machines). (N.B. Setting both include paths is essential to get any compile to work properly!)
    - (c) Press *OK* button to finish settings.

WARNING! STEPS 3 and 4 must be set in the sequence shown above. In particular if a change between *Release* and *Debug* is made it is necessary to set the *INCLUDE* paths again.

5. Under *Project* select *Add to Project* which causes a pop-up window to appear. Select *Files* which will pop-up another window called *Insert files into project*. Use the *Look in* window to select the folder where source programs are located and find the *feap* folder. The select *ver82* (double click on folder button will change path), followed by *contact* and then *main*. If *Files type* window is set to *Fortran files (\*.for, \*.f90,...)* all the files to be selected will appear in the large window. To select all files place mouse cursor over last file in folder and while holding the "Shift" key press the left mouse button. All files should now be highlighted. Press *OK* button to have highlighted files placed in project.
 

N.B. Instead of using the *Look in* window to find directories, it is possible to use the *Up one level* button to traverse the folder structure to locate where source files are located.
6. Repeat step 5 for all the source folder names in *contact* (i.e., *ptpnd*, etc.). Repeat for all subdirectories in *element*. Finally, load the files in the *plot*, *program*, *user* and *windows* directories. After all files (except `feap82.f` are loaded into the library:
7. Under *Build* tab select *Build lib82.lib* (or name you selected for this project or *Rebuild all*).

Compiler should process each file in the project and finish with a statement: "lib82.lib - 0 error(s), 0 warning(s)". If errors are present changes are necessary. First thing to ensure is that the path to the INCLUDE files is properly set (see step 4. above).

At this stage the library "lib82.lib" for the *FEAP* program has been built. It is now necessary to build the final executable program.

### 2.2.2 Build of Executable

The following steps may be used to build an executable for the *FEAP* program:

1. Under *File* select *New*.
  - (a) Under *Projects* tab select *Fortran Standard Graphics* or *QuickWin Application*.
  - (b) In *location* window path to location for build files should still be set for the library build. This is ok, but can be changed if you wish (recommend no change for this). The path must exist, if not use standard Windows steps to create the folder before doing this step.
  - (c) In *Project name* assign a program name (e.g., feap).
  - (d) Press *OK* button to start (N.B. small upper window should now have the notation *Workspace 'feap'*).
  - (e) New pop-up window gives choice between a *QuickWin* and a *Standard Graphics* mode. Select *QuickWin* and then press *Finish*.
2. Repeat steps 3 and 4 above which are now applicable to this project. (e.g., must set *Release* or *Debug* mode and path for INCLUDE files).
3. Under *Projects* tab select *Settings*, followed by the *Link* tab. In *Category* window select *Input*. In *Ignore libraries* window add ,libc.lib (with no blanks before the ","). N.B. Leave the existing entry (dfconsul.lib).
4. Use *Project* tab and select *Add to Project*. Then select *Files* and select the folder *Main* (see step 5 above). Add the main program file 'feap82.f' to the project. (If you do not want to include the graphics option also add the file 'contact.f').
5. Use *Project* tab and select *Add to Project*. Select *Files* tab and go to folder where library "lib82.lib" is located. This is the path you set in the first build followed by the name of the library (e.g., "lib82") and either *release* or *debug* depending on which you built. Nothing will appear in the main window until a selection

is made in the *Files of type* window is set to: *Library Files (lib)*. It may be necessary to scroll to find this or just enter "l" in the window and scrolling will occur automatically.

Add the `lib82.lib` to the project by placing the mouse over the name in the window and double clicking.

6. Under *Build* tab select *Build feap.exe* (or name you selected for this project or *Rebuild all*. Compiler should process each file in the project and finish with a statement: "feap.exe - 0 error(s), 0 warning(s)". If errors are present changes are necessary. First thing to ensure is that the path to the INCLUDE files is properly set (see Step 4. in the instructions for building the libraries).

Program is ready to use. The executable will be placed in the *release* or *debug* directory where the build of the executable was designated (see Step 3 and 4 in Section 2.2.2). It is usually desirable to place an executable icon on the 'Desktop'.

### 2.2.3 Alternate Windows graphics form

An alternate to the graphics structure may be built by using the source files in the directory `window2` instead of those with the same names located in the `windows` directory. Note that the remaining files in `windows` must still be included. The alternate form uses the source files from `window2` when building the executable instead of the files with the same name that are in the `windows` directory (the files to be included are: `pfullscr.f`, `popen.f`, `plstrt.f`, `plstdos.f`, `pwopn.f`).

### 2.2.4 Running FEAP

After a successful installation step the *FEAP* program is ready for use. The program may be run in two modes:

1. From a command line in a "Command prompt" window. In this case it is convenient to place a "bat" file (e.g., `feap82.bat`) in a directory located on the system PATH. This file has the structure:

```
c:\fullpath\feap\ver75\build\release\feap
```

where it is assumed the executable resides in the directory `build` and is a release version. The program may now be executed by giving the command

```
feap
```

in any directory.

2. From an icon reached by traversing the directories to the location where the executable resides after the build. For convenience the icon may be placed on the 'Desktop' and executed there. A pop-up window will appear to locate the desired input file (see User Manual for preparing this file).

## 2.3 Installation with no graphics

For any system in which there is neither support for X11 nor graphics based on the Visual Fortran graphics interface an executable may be build without plotting capability. While this is not desirable it still allows for problems to be solved using the *FEAP* system of other solution options.

To build an executable, first build a library (or archive) module that contains the following routines:

Directory	Files
contact	All files in all subdirectories
element	All files in all subdirectories
plot	All files
program	All files except: outary.f pgetd.f pprtd.f
unix	All files except: doargs.f pinitm.f pstart.f setmem.f tinput.f and all .c files
user	All files
patch	Files: doargs.f pstart.f tinput.f
patch/memory	Files: outary.f pgetd.f pinitm.f pprtd.f setmem.f

Build the main program by including the following files include directory to: Directory Files main feap82.f gdx11.f pplotf.f Library your library or archive file built above

This version does not include any graphics and also uses a *blank common* to store all the arrays. The maximum size is set in the subprogram `pinitm.f` in the `patch/memory` directory (currently it is 80,000,000 integer words (N.B. It is assumed that 32 bits are used for integers. On 64bit systems it is necessary to change `integer` to `integer*8` in `pinitm.f`.)