Compiling UPFs

Sheldon Imaoka

Memo Number: STI0901C
ANSYS Release: 12.1
Original Date: February 7, 2009

1 Introduction

ANSYS User Programmable Features (UPFs) are quite powerful means of customizing ANSYS beyond the scripting capabilities of APDL (ANSYS Parametric Design Language). Writing one's own constitutive models or complex loading functions or even creating a user-defined element are possible with UPFs.

Because some users may find it challenging to get started using UPFs, this memo attempts to discuss general issues related to compiling and linking a customized version of ANSYS for Windows and Linux environments.[1]

2 Getting Started

Before getting started, the user should refer to two important ANSYS documents. The *ANSYS Installation and Licensing Documentation* is accessible from an ANSYS installation or from the ANSYS Customer Portal.[2] Within this document are the *ANSYS, Inc. Windows Installation Guide* and the *ANSYS, Inc. UNIX/Linux Installation Guide*, both of which list the required compilers for each operating system. The user should keep in mind that only the versions of the compilers noted in these documents are supported and have been tested.

The second important reference is the *Programmer's Manual for ANSYS*.[3] This manual contains very useful information on UPFs and support-

[1]UPFs are also available on IBM AIX, Sun Solaris, SGI IRIX, and HP-UX systems.

[2]`http://www1.ansys.com/customer/`

[3]From ANSYS 12.0.1 onwards, the *Programmer's Manual for ANSYS* is included as part of the regular documentation of an ANSYS installation. For ANSYS 11.0 and prior versions, the manual must be downloaded from the ANSYS Customer Portal.

ing routines, so the user should review this document prior to attempting to use any UPFs.

Files for the creation of UPFs are not installed by default, so the user must verify that the customization files are included on their computer. In the ANSYS installation directory, a folder called "custom" should be present on Windows[4], whereas on Linux, a subdirectory called "customize" should exist. If the user does not find such a directory, the user should contact their IT administrator and reinstall ANSYS, being sure to specify that the customization files also be installed.

Lastly, the user should already be familiar with programming in Fortran, but having a good Fortran programming book on hand can serve as a helpful reference.

3 Overview of Process

User Programmable Features typically involve writing subroutines in Fortran. In ANSYS, Fortran 90 is the supported language, and there are many examples in the "custom" or "customize" directory for each of the supported user subroutines.

The general steps are as follows:

1. Assess whether the desired feature/capability is best implemented by APDL or by UPFs.
2. If UPFs are needed, check Reference [7] to determine which subroutine is best suited for the task.
3. Find the example subroutine in the "custom" or "customize" directory and modify it accordingly.
4. With the supported compiler listed in Reference [6], compile subroutine to create the object file.
5. With the supported linker, link the compiled object file and supplied libraries to create a custom ANSYS executable.[5]

[4]In ANSYS 11.0 and prior versions, a single sub-folder called "custom" will be present. In ANSYS 12.0.1 and later revisions, both "custom" and "customize" folders will be present.

[5]On Linux systems, it is possible to create a shared library instead of an executable. See Section 5.9 in Ref [7] for details. On Windows systems, a dynamic-link library (DLL) can be created for many popular UPFs from ANSYS 12.1 onwards.

6. Run the customized ANSYS executable. In the ANSYS session, activate the user-programmable feature using the relevant ANSYS command.

The following subsections will discuss the above process in more detail.

3.1 APDL vs. UPFs

While this may sound trivial, there have been many situations where users have mistakenly thought that UPFs were required for tasks that are easier to perform using APDL. While APDL can be thought of as a scripting language that may be slower to execute than compiled code, APDL has vector functions that speed up tasks considerably. Moreover, changing/adjusting APDL input files is much easier than modifying subroutines and recompiling/relinking. Hence, the user should consider whether or not the desired functionality can be achieved with APDL, as that would generally be a much easier, quicker approach than implementing UPFs.

3.2 Determining the Right Subroutine to Use

In Reference [3], there are many subroutines to choose from, and some may seem to provide duplicate functionality with other subroutines. There are some legacy subroutines available for users who may wish, for one reason or another, to stick with older elements or older features. The "newer" subroutines are typically associated with the current-technology elements.[6]

For example, to define one's own nonlinear constitutive model, consider using `USERMAT.F` instead of `USERPL.F`.

3.3 Compiling and Linking UPFs

Example UPFs are provided in the "customize\user" subdirectory.[7] Also with the example UPFs is a script called `ANSCUSTOM` (`ANSCUST.BAT` on Windows) that facilitates linking/compiling. Before running that script, it is important to note `ANSCUSTOM` on any operating system needs to know (a) where the compiling/linking executables reside, (b) where system "include" files are located, and (c) where "libraries" are located. These are usually

[6]Current technology structural elements include `LINK180`, `BEAM188-189`, `PIPE288-289`, `ELBOW290`, `SHELL181/281`, `SHELL208-209`, `SOLSH190`, `SOLID185-187`, `SOLID272-273`, `SOLID285`, and `PLANE182/183`.

[7]On Windows for ANSYS 11.0 and prior, this was in the "custom" subdirectory.

achieved through the definition of the PATH, INCLUDE, and LIB environment variables.

All of the user subroutines are in a single folder, as noted above. However, the user does *not* need to compile all of the supplied subroutines — for example, if a user is only interested in implementing a custom creep law, only `usercreep.F` needs to be copied, compiled, and linked. The author prefers to copy the necessary files to a separate location, although in ANSYS 12.0.1 onwards, the user may wish to compile in the "custom\user\platform" subdirectory directly.

3.3.1 Setting up Compiling on Windows for ANSYS 11

On Windows, one typically installs the Microsoft compiler first, then the Intel compiler. When a compatible Microsoft compiler has been installed, the Intel Fortran batch file (`IFORTVARS.BAT`) is updated to include the path to the Microsoft linker, which facilitates the compiling process.

For ANSYS 11.0 SP 1 on 64-bit Windows, the supported compiler for Windows is *Intel Fortran 9.1* while, for ANSYS 11.0 SP 1 on 32-bit Windows or 11.0 (original release) for both 32-bit and 64-bit Windows, the supported compiler is *Intel Fortran 8.1* — Reference [4] for ANSYS 11.0 SP1 should be reviewed for the compiler change. Please note that when a user purchases an Intel compiler, they can register on the Intel Development site[8]. Once logged in, the user may not only obtain updates to the current Intel Fortran version but also obtain older versions, such as versions 8.1 or 9.1.

The supported Microsoft compilers are listed in References [2] and [4], as mentioned earlier. It is worth noting that *Microsoft Visual C++ 2005 Express Edition*[9] or *Platform SDK*[10] are also known to work for linking purposes, and these products are available for free. However, the author has not attempted to try all different versions of these products, so no guarantee is made that these unsupported software will work for creating customized ANSYS executables. (Microsoft has renamed *Platform SDK* to *Windows SDK* for current releases, and older *Platform SDK* releases should still be available from Microsoft's website.)

After installation of both Microsoft and Intel products (in that order), from the "Start" menu, one should find a shortcut to the "Build Environment for Fortran" in the "Intel Software Development Tools" submenu. Right-click and select "Properties" to see the path of the shortcut — it

[8] `https://registrationcenter.intel.com/RegCenter/`

[9] `http://www.microsoft.com/downloads/details.aspx?displaylang=en\&FamilyID=7b0b0339-613a-46e6-ab4d-080`

[10] `http://www.microsoft.com/downloads/details.aspx?displaylang=en\&FamilyID=0baf2b35-c656-4969-ace8-e4c`

should point to a batch file called `IFORTVARS.BAT`. Open this batch file in a text editor, and one should see a `call` statement that calls the Microsoft Visual C++ environment batch file `VCVARS32.BAT` or `VCVARSALL.BAT` (On Windows systems using the *Platform SDK*, the called file is actually `SETENV.CMD`). If this line is commented out, one may be able to replace it with a call to the appropriate *Microsoft Visual Studio* batch file.

Once this is done, the `IFORTVARS.BAT` batch file can be called or run to set up the correct PATH, LIB, and INCLUDE environment variables for the Intel Fortran compiler and Microsoft linker. (While a user may define these environment variables at the system level, the author does not recommend doing this since that would prevent multiple compiler versions to be installed and usable on the same PC.)

In the ANSYS installation directory, find `ANSCUST.BAT` in the "custom\user" folder. Open this in a text editor, and in the second line, add the DOS command `call <...>\ifortvars.bat`, where `<...>` should be replaced by the full path of the `IFORTVARS.BAT` batch file. If spaces are present, enclose the full path in double-quotes. Also, the author prefers to add a `pause` command at the very end of the `ANSCUST.BAT` file. That way, one can double-click on the `ANSCUST.BAT` file, and the DOS window will not disappear, allowing the user to determine whether compiling was successful or not.

An alternative to modifying the batch files through the process described above is to (1) open the appropriate Microsoft build environment command prompt, (2) `cd` to the directory of the Intel Fortran compiler and execute `IFORTVARS.BAT`, then (3) `cd` to the directory containing the subroutines and run the `ANSCUST.BAT` script. This manually performs the steps described earlier.

The files in the ANSYS "user\platform" subdirectory that are required for compiling are as follows:

- `ANSCUST.BAT` (or, for Large Memory version on 64-bit Windows, `ANSCUST_LM.BAT`)
- `ansyslarge.def` and `ansyssmall.def`
- `MAKEFILE` (or, for Large Memory version on 64-bit Windows, `MAKEFILE_LM`)
- `*.F` subroutine(s) to compile

The author prefers copying the necessary files listed above to a separate directory for compiling purposes.

Once this has been set up, one may double-click on the `ANSCUST.BAT` batch file to create the new, customized ANSYS executable.

Once compiling is successful, the user should see several files created:

- `*.obj` are compiled object files for each compiled `.F` file
- `f_comp.log` (and, if `*.c` files are used, `c_comp.log`) contain output from the compiler, which is useful to review error messages that may appear
- There will be several `ANSYS.*` files, the most important of which are the `ANSYS.exe` customized executable and `ANSYS.exe.manifest` manifest files. The other files, such as `ANSYS.lib` and `ANSYS.exp` and `ANSYS.map` are not required, although they can just be left in the directory.

3.3.2 Setting Up Compiling on Windows for ANSYS 12

In addition to items noted in Section 3.3.1, there are a few changes in ANSYS 12.0.1 and 12.1 worth discussing.

The first point is that the "custom" subdirectory has been split — the "custom" folder contains platform-dependent files while the "customize" directory has the example subroutines and files that are generally platform-independent. The user may wish to compile directly in the "custom\user\platform" folder by copying relevant `*.F` files to this location and running `ANSCUST.BAT`.

Secondly, in version 12.0.1 onwards, there is no separate "large memory" version of ANSYS. The "regular" version of ANSYS on 64-bit Windows supports large memory ($>$ 16 GB) for the sparse direct solver. Consequently, there is no longer a need for a separate `ANSCUST_LM.BAT` batch file.

The user is strongly encouraged to use the listed, support compilers. However, the author has found that the combination of *Microsoft Visual C++ 2005 Express Edition*[11] and *Microsoft Platform SDK*[12] or *Microsoft Windows SDK*[13] works fine, with a few caveats — please see Subsections 4.1.7 and 4.1.8 for details.

3.3.3 Setting Up Compiling on Linux

For Linux, the user should install the supported compilers listed in Reference [2]. As with the case in Windows, the proper PATH, LIB, and IN-

[11] `http://www.microsoft.com/downloads/details.aspx?displaylang=en\&FamilyID=7b0b0339-613a-46e6-ab4d-08`

[12] `http://www.microsoft.com/downloads/details.aspx?displaylang=en\&FamilyID=0baf2b35-c656-4969-ace8-e4`

[13] `http://www.microsoft.com/downloads/details.aspx?displaylang=en\&FamilyID=e6e1c3df-a74f-4207-8586-71`

CLUDE environment variables need to be defined. For the *Intel Fortran Compiler for Linux*, both C-shell and Bourne-shell scripts are available to set this up with names of `ifortvars.csh` and `ifortvars.sh`, respectively. Add the command to `source` this file in your shell configuration file (e.g., `.tcshrc` for tcsh or `.bashrc` for bash).

If the user does not have permissions to modify the "customize/user" subdirectory, copy its contents to a folder in the home directory. From there, the `ANSCUSTOM` script can be executed simply by typing its name (or you may need to use `./ANSCUSTOM` and verify that the script is executable). It is worth noting that only the `ANSCUSTOM` script and desired user subroutine need to be copied to the user's directory to create a customized executable, so the user does not have to copy and compile all files of the ANSYS "user" subdirectory.

After the script completes successfully, the user should see an `.o` object file for each subroutine compiled, `link.err` and `link.out` diagnostic files (on Linux, the Intel Fortran `ifort` script also performs the linking), and the customized ANSYS executable called `anscust.e121`. The user can view the contents of the `link.err` and `link.out` files to see if any problems occurred.

3.4 Running the Customized ANSYS Executable

After the compiling and linking process is successful, users can use the "Mechanical APDL Product Launcher" (`launcher121` command on Linux) to run the customized executable. On the "Customization/Preferences" tab of the "Mechanical APDL Product Launcher", specify the full path of the "Custom ANSYS Executable". *On Windows, it is also important to note that the ANSYS "bin\platform" subdirectory must be included in the system PATH environment variable for the customized executable to find the relevant DLLs.*

For users comfortable with command-line execution, the syntax is `ansys121 -custom <path>` with any additional arguments included. The `<path>` should be the absolute or relative path to the customized executable. On Windows, the customized executable is called `ansys.exe` whereas on Linux, it is called `ansyscust.e121`.[14]

After launching ANSYS, to verify that the customized version is being used, refer to the Output Window or Output File for the following line:

[14] In ANSYS 9.0 and prior, Linux users needed to use the `ansyscust90` script instead of `ansys90` to invoke the customized executable. Also, in version 9.0 and prior, the customized executable was named `ansys.eVVt` instead of `ansyscust.eVVV`, where `VV` is the version number.

```
 Note - This ANSYS version was linked by Licensee
```

which verifies that the customized rather than standard version is being used.

4 Special Topics

4.1 Troubleshooting

Below are some troubleshooting tips.

4.1.1 I run ANSCUST.BAT but nothing happens

Follow the instructions given in Section 3.3.1. Check the DOS window to see what error message(s) may be present. If error messages such as `'ifort' is not recognized as an internal or external command, operable program or batch file.` appear, this means that the `IFORTVARS.BAT` file was not called properly. If errors such as `'nmake' is not recognized as an internal or external command, operable program or batch file.`, this means that the *Microsoft Visual C++* or *Visual Studio* environment was not defined correctly. Lastly, if a `.lib` file is missing, search your computer for that `.lib` file. If the file is present, check to see its location and whether or not the LIB environments may be defined correctly; if the file is not present, most likely the correct Microsoft compiler is not installed on the system.

4.1.2 I get an "unresolved external symbol ___powr8i4 referenced" error during compiling

On Windows systems with ANSYS 10.0 or 11.0 (original release), this may appear when using Intel Fortran 9.1 or higher version and a power function (e.g., `x**y`) is present in the subroutine. Either redefine the equation or use the supported version (Intel Fortran 8.1).

Conversely, if running ANSYS 11.0 SP 1 on 64-bit Windows, a similar error appears if the user is compiling with Intel Fortran 8.1 since the compiler requirement has been upgraded for 11.0 SP 1 per Reference [4]. Upgrade to Intel Fortran 9.1 to resolve this problem.

4.1.3 When using a custom ANSYS executable, I try to save a hardcopy to file but ANSYS aborts

This problem may occur if you are using *Microsoft Visual Studio 2005 Express Edition* or *Microsoft Visual Studio 2005*, where an incompatible graphics library is present. After solving the simulation with the custom executable, one can postprocess state variables and other quantities with the standard ANSYS version, allowing users to save graphics images if unable to do so with the customized ANSYS version.

4.1.4 When compiling version 9.0 or 10.0 of ANSYS on Linux, I get an "undefined reference to 'glPolygonOffsetEXT"' error

This may be due to missing XFree86 development libraries (required packages are listed in Reference [2]). Many newer Linux distributions use X.Org rather than XFree86 for the X server implementation, and the appropriate OpenGL libraries may not be installed. (On Ubuntu or Debian, the required package may be called `libgl1-mesa-swx11-dev`.)

4.1.5 When compiling on Linux, I get messages related to "undefined reference to 'gluXXXXXXX' error

The "glu" references are related to Mesa 3D (OpenGL implementation on Linux) development files. Check Ref [2] for the Mesa packages that need to be installed on the system. Note that an older Mesa library may possibly need to be installed, and be sure to install the development files (often with a "-dev" or "-devel" prefix).

4.1.6 When linking a shared library on Linux, I get the error "relocation R_X86_64_PC32 against undefined symbol 'log' can not be used when making a shared object; recompile with -fPIC"'

As the error message indicates, the "-fPIC" (or "-fpic") argument must be supplied. This may be due to using a newer version of the Intel Fortran compiler on Linux, where the use of "-KPIC" or "-Kpic" has been removed. In the `ANSUSERSHARED` script, search for occurrences of the `ifort` command, and change the argument of "-Kpic" to "-fPIC" for the appropriate platform for the compiling case.

4.1.7 When compiling on Windows for ANSYS 12.0.1, I get an error about "vcomp.lib" not being found

In ANSYS 12.0.1 onwards, many non-solver areas of ANSYS (Mechanical APDL) have been parallelized using OpenMP for Shared-Memory ANSYS.

The `vcomp.lib` file is the OpenMP library (similarly, `vcompd.lib` is the debug version of the OpenMP library).

If you use *Microsoft Visual C++ 2005 Express Edition* with *Microsoft Windows Server 2003 Platform SDK* for linking, the `vcomp.lib` library is not included. You can try installing and using *Microsoft Windows SDK* (later releases of *Platform SDK* are renamed to *Windows SDK*) instead, as these include `vcomp.lib`.

In ANSYS 12.1, the `vcomp.lib` library is supplied (located in the `custom\lib` subdirectory, so *Microsoft Visual C++ 2005 Express Edition* with *Microsoft Windows Server 2003 Platform SDK* should work fine.

4.1.8 When compiling on Windows for ANSYS 12.1, I get an error about "bufferoverflowU.lib" not being found

When using *Microsoft Visual C++ 2005 Express Edition* with either *Microsoft Windows SDK for Vista* or *Microsoft Windows SDK for Windows Server 2008*, the library `bufferoverflowU.lib` does not seem to be included.

The older *Microsoft Windows Server 2003 Platform SDK* does seem to include this library, so install that version instead (or use the regular *Microsoft Visual Studio 2005 Professional Edition*, which is the supported compiler version).

4.1.9 When compiling on Windows for ANSYS 12.0.1 and 12.1, I get an "unresolved external symbol" error

For reference, the full text of the error is as follows:

> ansys1.lib(FullHarmonicVT.obj) : error LNK2019: unresolved external symbol "__declspec(dllimport) private: static void __cdecl std::locale::facet::facet_Register(class std::locale::facet *)" (__imp_?facet_Register@facet@locale@std@@CAXPAV123@@Z) referenced in function "class std::ctype<char> const & __cdecl std::use_facet<class std::ctype<char> >(class std::locale const &)" (??$use_facet@V?$ctype@D@std@@@std@@YAABV?$ctype@D@0@ABVlocale@0@@Z) ansys1.lib(C_ItfAnsFreqSweep.obj) : error LNK2001: unresolved external symbol "__declspec(dllimport) private: static void __cdecl std::locale::facet::facet_Register(class std::locale::facet *)" (__imp_?facet_Register@facet@locale@std@@CAXPAV123@@Z) ansys1.lib(FullHarmonicVT.obj) : error LNK2019: unresolved external symbol "__declspec(dllimport) public: static unsigned int

```
__cdecl std::ctype<char>::_Getcat(class std::locale::facet const *
*)" (__imp_?_Getcat@?$ctype@D@std@@SAIPAPBVfacet@locale@2@@Z)
referenced in function "class std::ctype<char> const & __cdecl
std::use_facet<class std::ctype<char> >(class std::locale const
&)" (??$use_facet@V?$ctype@D@std@@@std@@YAABV?$ctype@D@0@ABVlocale@0@@Z)
ansys1.lib(C_ItfAnsFreqSweep.obj) : error LNK2001: unresolved
external symbol "__declspec(dllimport) public: static unsigned
int __cdecl std::ctype<char>::_Getcat(class std::locale::facet const
* *)" (__imp_?_Getcat@?$ctype@D@std@@SAIPAPBVfacet@locale@2@@Z)
ANSYS.exe : fatal error LNK1120: 2 unresolved externals
```

This can occur if the user is using *Microsoft Visual C++ 2008 Express Edition* for linking. This version is not supported at ANSYS 12.0.1 or 12.1 — please use *Microsoft Visual Studio 2005 Professional Edition* (officially supported) or *Microsoft Visual C++ 2005 Express Edition*.

4.2 Creating a Shared Library on Linux and Windows

On Linux systems, instead of compiling an executable, a shared library may be created instead. The `ANSUSERSHARED` script should be run in a separate directory with the appropriate `*.F` files present. After the shared library is successfully created, the object file(s) `*.o` as well as `userlib.a` and shared library `libansuser.so` should exist in the directory. Error messages, if any, may be found in the file `shared.log`.

Per the instructions given during the execution of the `ANSUSERSHARED` script, set the `ANS_USER_PATH` environment variable to the directory where `libansuser.so` resides.[15] By executing ANSYS in a normal fashion, one will see the printout

```
 User Link path (ANS_USER_PATH): /home/username/directory
```

in addition to the

```
 Note - This ANSYS version was linked by Licensee
```

message, indicating a customized version is being run using a shared library.

On Windows systems, creating a dynamic-link library (DLL) is possible starting from ANSYS 12.1 for certain UPFs. Run the `ANSUSERSHARED.BAT` batch file to create the DLL (be sure to hit the "Enter" key to exit out of the batch file). Set the environment variable `ANS_USER_PATH` in the System Control Panel to the directory where the DLL(s) resides.

[15]For Bourne shells, use `export ANS_USER_PATH=/path/to/dir` while for C-shell environments, use `setenv ANS_USER_PATH /path/to/dir`.

4.3 Special Versions of ANSYS

On 64-bit Windows for ANSYS 11.0, the `ANSCUST_LM.BAT` batch file can be used to create a customized, large-memory version of ANSYS. The large-memory version of ANSYS is required only for accessing more than 16 GB of physical RAM with the sparse direct solver, and its usage is discussed in Section 3.3 of Reference [1]. The associated makefile is `MAKEFILE_LM`. The procedure is the same as covered in Section 3.3.1. In ANSYS 12.0.1 onwards, the large memory version is now the "default" version of ANSYS, so there is no special `ANSCUST_LM.BAT` required anymore.

On Linux systems with ANSYS 10.0 and 11.0, as well as Linux and Windows systems with ANSYS 12.0.1 onwards, creating a customized Distributed ANSYS version is possible.[16] On Linux, during the `ANSCUSTOM` script execution, the user is asked whether a Distributed ANSYS version is required — if the user chooses to compile the distributed version, `ansMPIComm.o` and `ansMPIFortranCall.o` files will be compiled, and the resulting executable will be named `ansyscustdis.e110`.

4.4 Cross-Compiling

Although more of an "advanced" topic, it is worth noting that cross-compiling is also possible. For example, on a 64-bit Windows PC, a user may create a customized executable for 32-bit Windows, although it is not possible to create a 64-bit Windows executable from a 32-bit Windows system. To cross-compile, the user (a) must have a compiler that supports cross-compiling and (b) needs to have that platform's "custom" directory available.

To cross-compile 32-bit Windows on 64-bit Windows, do the following:

1. Install the appropriate cross-compilers. On 64-bit Windows for ANSYS 11.0 SP1, the 32-bit versions of *Intel Fortran 9.1* and *Microsoft Platform SDK SP1* are usually installed by default during a typical installation process.

2. Obtain the ANSYS "custom" directory for a 32-bit Windows installation, and copy it to the 64-bit Windows' ANSYS installation directory. Note that platform-specific files are located in an "intel" or "winx64" subdirectory, so it is possible to simply copy over the contents. Please note, however, that it is not good practice to mix 11.0 original release files with those of 11.0 SP 1.

[16] Details on Distributed ANSYS can be found in Reference [5].

3. (Optional) If 32-bit testing will need to be performed on the 64-bit PC, also copy the "intel" subdirectories of the "bin" and "lib" directories.

4. Modify the 32-bit version of the `ANSCUST.BAT` file as noted in Section 3.3.1. Be sure that the `IFORTVARS.BAT` of the 32-bit version of the Intel Fortran compiler is being referenced.

5. Run `ANSCUST.BAT`. Both the Microsoft and Intel compilers should indicate that a 32-bit version is being run. This should allow the user to compile a 32-bit `ANSYS.EXE`.

6. (Optional) To perform tests, the easiest method is to create a DOS batch file in the same folder as the 32-bit customized ANSYS executable. Add `set ANSYS_SYSDIR=intel` to set the platform to 32-bit Windows, and include the appropriate DLLs with `set PATH=%ANSYS110_DIR%\bin\%ANSYS_SYSDIR%;%PATH%`. Then use `ansys110 -g -custom .\ansys.exe` in the file. Double-click on this batch file and verify in the Output Window that this the platform is "INTEL NT" and that it is a customized version (look for the "linked by Licensee" note).

On 32-bit and 64-bit Linux systems, the general procedure is similar to above. On the 64-bit Linux PC, the user should ensure that they have the 32-bit versions of the ANSYS customization files, along with the 32-bit version of the appropriate compiler and development files.

4.5 Using Unsupported Compilers

Only the compilers listed in Ref [2] should be used for creating a customized ANSYS executable. Using different versions of supported compilers is generally OK, although there are some exceptions, a couple of which have been noted in Section 4.1.

The GNU compilers, while attractive because they are free, are generally not suitable for this use. The Fortran compiler in GCC 3.x is `g77` and does not support many required features and was not usable. The author has had mixed success in the past using `gfortran` in GCC 4.x but has since abandoned any attempts to use the GNU Fortran compilers.

5 Conclusion

This memo attempted to cover some basic items related to creating one's own custom ANSYS executable on Windows and Linux.

When the ANSYS Customization Files are selected during the ANSYS installation process, example subroutines are made available to the user, providing a useful reference for the implementation of one's own UPF. ANSYS also supplies scripts to automate the compiling and linking process. This memo attempted to supplement existing documentation to provide the user with additional details and tips on creating customized ANSYS executables on 32-bit and 64-bit Windows and Linux platforms.

Revisions to this Document

- STI0901A (February 13, 2009): Added Section 4.2 on shared library for Linux, Section 4.3 on special versions, added some additional error messages
- STI0901B (April 13, 2009): Had earlier notes in Sections 3.3.1 and 4.1 indicating that ANSYS 11.0 SP 1 on Windows required Intel Fortran 9.1. Reference [4] states that only 64-bit Windows at ANSYS 11.0 SP 1 requires Intel Fortran 9.1, so the above sections were corrected.
- STI0901C (January 9, 2010): Added information for ANSYS 12.0.1 and ANSYS 12.1.

References

[1] ANSYS, Inc. *ANSYS 11.0 Basic Analysis Guide*, 2007.

[2] ANSYS, Inc. *Installation and Licensing Documentation*, 2007.

[3] ANSYS, Inc. *Programmer's Manual for ANSYS*, 2007.

[4] ANSYS, Inc. *Release 11.0 Service Pack 1 Notes*, 2007.

[5] ANSYS, Inc. *Distributed ANSYS Guide*, 2009.

[6] ANSYS, Inc. *Installation and Licensing Documentation*, 2009.

[7] ANSYS, Inc. *Programmer's Manual for Mechanical APDL*, 2009.

Sheldon's ansys.net Tips and Tricks

Sheldon's ansys.net Tips and Tricks are available at the following URL:

`http://ansys.net/sheldon_tips/`

Please remember that, with each release of ANSYS, new features and techniques may be introduced, so please refer to the ANSYS documentation as well as your local ANSYS support office to verify that these tips are the most up-to-date method of performing tasks.

Disclaimer: the author has made attempts to ensure that the information contained in this memo is accurate. However, the author assumes no liability for any use (or misuse) of the information presented in this document or accompanying files. Please refer to ansys.net for the latest version of this document. Also, this memo and any accompanying input files are not official ANSYS, Inc. documentation.

ANSYS Training

ANSYS, Inc. as well as ANSYS Channel Partners provide training classes for ANSYS, Workbench, CFX, FLUENT, ANSYS LS-DYNA, AUTODYN, ASAS, AQWA, TAS, and ICEM CFD products. Information on training classes and schedules can be found on the following page:
`http://www.ansys.com/services/ts-courses.asp`

ANSYS Customer Portal

Customers on active maintenance (TECS) can register for a user account and access the ANSYS Customer Portal. Here, browsing documentation, downloading software (including service packs), and submitting technical support incidents are possible: `http://www1.ansys.com/customer/`

XANSYS Mailing List

The XANSYS mailing list is a forum for exchanging ideas, providing and receiving assistance from other users, and general discussions related to ANSYS and Workbench. (Note that it is recommended to contact your local ANSYS support office for technical support.) You can obtain more information by visiting the following URL: `http://www.xansys.org/`