Compiling FAST v8.08.00c-bjj

Bonnie Jonkman  
National Renewable Energy Laboratory  
June 30, 2014

# Introduction

FAST v8.08.00c-bjj is distributed with several options for compiling the source code. It contains a Visual Studio project file, a Windows® batch file for Intel Fortran, and a makefile for gfortran. The FAST archive contains all of the source files necessary to compile, and the compiling tools are set up with the correct paths to those files. The tools also set up all the compiling options required to compile FAST.

The compiling tools distributed with FAST v8.08.00c-bjj were created for Windows® systems. However, they can be adapted for use on non-Windows® systems. Please see “Compiling on Linux or Mac” at the end of this document for some suggestions.

We have developed these instructions with the following two compilers:

* Intel® Visual Fortran (IVF) Composer XE 2011, version 12.1.3.300  
  Intel Math Kernel Library (MKL) 10.3 Update 9  
  Microsoft Visual Studio 2010 Shell
* gfortran/gcc version 4.6.2  
  mingw32

# Compiling with Intel Fortran in Visual Studio

Open “FAST\_Project.vfproj” in the Compiling\VisualStudio directory of the FAST v8 archive. This should open a Solution Explorer window that lists all of the source files and Registry input files for FAST v8.08.0cc-bjj. See Figure 1.

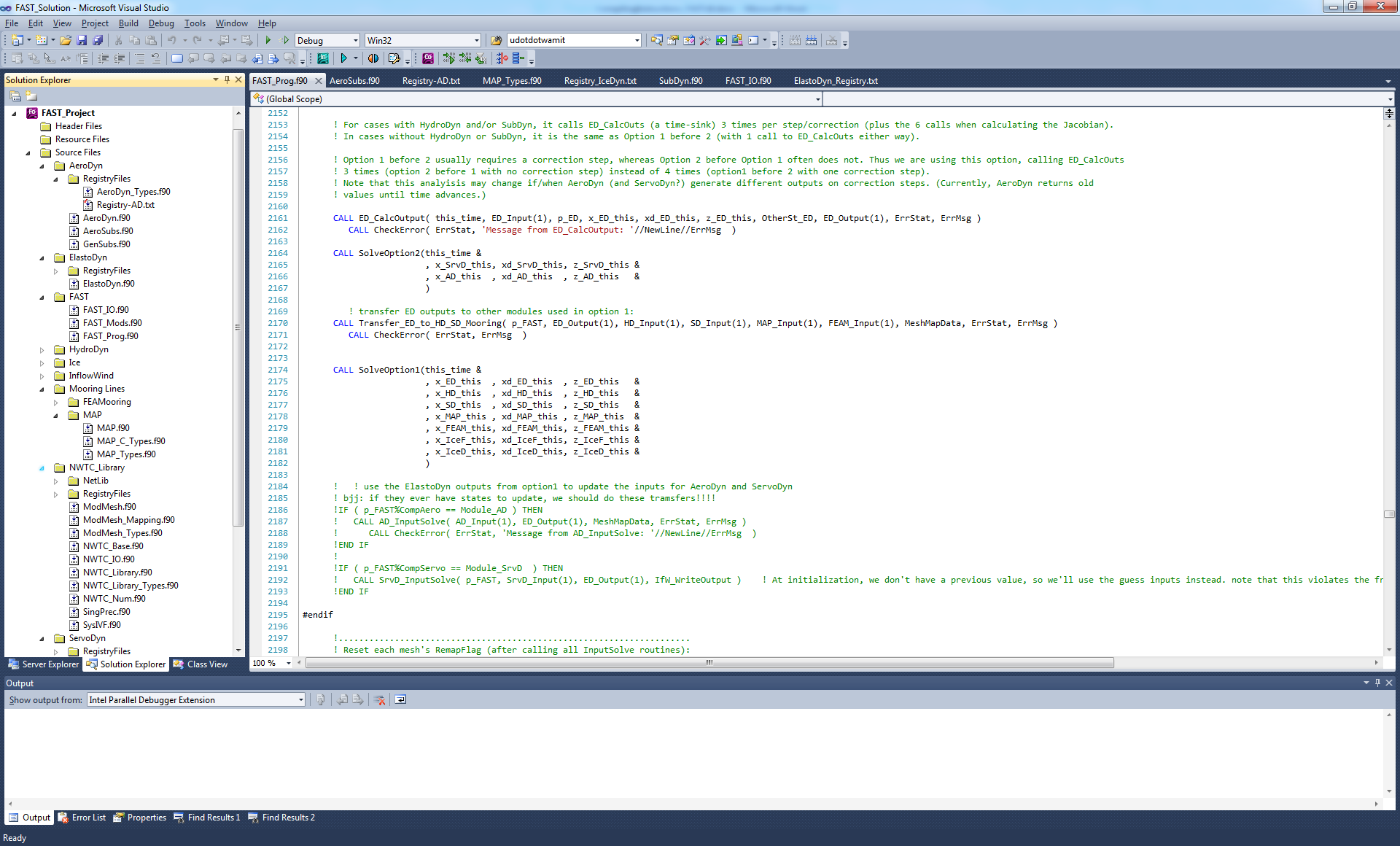


Figure 1. Source Files in the FAST Visual Studio Project

Choose either Debug or Release configuration from the Configuration manager (Debug is the default), and then select “Build Solution” from the “Build” menu. This will run the FAST Registry to produce the \*\_Types.f90 files needed in the project and then will compile and link the project. When it has successfully completed, it will place an executable called “FAST\_dev\_Win32.exe” (Release mode) or “FAST\_dev\_debug\_Win32.exe” (Debug mode) in the FAST\bin folder.

When you compile the first time, a box will probably pop up to ask you to save the solution file. Select “ok” to save the solution file in the same directory as your project file.

Note that sometimes the Registry will create a \*\_Types.f90 file that does not get compiled when building the solution. If that happens, you will need to select “Build Solution” a second time, so the \*\_Types.f90 file will compile.

Note that this solution requires that you have the Intel® Math Kernel Library (MKL) to access LAPACK routines. MKL comes with newer versions of Intel® Visual Fortran (IVF), however some old versions of IVF required a separate license for MKL. If you have the Intel compiler without access to MKL, see the instructions at this web site: <http://icl.cs.utk.edu/lapack-for-windows/lapack/> for help in installing LAPACK and BLAS libraries for Release and Debug modes.

# Compiling with Intel Fortran from a Windows Command Prompt

FAST is distributed with a batch file called “Compile\_FAST.bat” that will compile the code using IVF.

Before using Compile\_FAST.bat, either you must modify variables in the section labeled “set compiler internal variables” or you must run Compile\_FAST.bat *only* from the compiler’s command prompt window.

## Setting Compiler Internal Variables

In the “Set Compiler Internal Variables” section, you make sure that the proper paths and environment variables are set for the compiler and linker. If this step is not done correctly, you will not be able to compile. The **blue** text shown in Figure 2 (copied from Compile\_FAST.bat) must be changed to reflect your compiler.

REM ----------------------------------------------------------------------------

REM set compiler internal variables

REM ----------------------------------------------------------------------------

REM You can run this bat file from the IVF compiler's command prompt (and not

REM do anything in this section). If you choose not to run from the IVF command

REM prompt, you must call the compiler's script to set internal variables.

REM TIP: Right click on the IVF Compiler's Command Prompt shortcut, click

REM properties, and copy the target (without cmd.exe and/or its switches) here:

CALL **"C:\Program Files (x86)\Intel\ComposerXE-2011\bin\ipsxe-comp-vars.bat" ia32 vs2008**

REM ----------------------------------------------------------------------------

REM -------------------- LOCAL VARIABLES ---------------------------------------

REM ----------------------------------------------------------------------------

SET ROOT\_NAME=FAST

SET COMPOPTS=/threads /O2 /inline:speed /traceback /Qzero /Qsave /real\_size:32 /assume:byterecl

SET LINKOPTS=/link

REM ----------------------------------------------------------------------------

REM ------------------------- LOCAL PATHS --------------------------------------

REM ----------------------------------------------------------------------------

REM -- USERS WILL NEED TO EDIT THESE PATHS TO POINT TO FOLDERS ON THEIR LOCAL --

REM -- MACHINES. NOTE: do not use quotation marks around the path names!!!! ---

REM ----------------------------------------------------------------------------

REM NWTC\_Lib\_Loc is the location of the NWTC subroutine library files

REM AeroDyn\_Loc is the location of the AeroDyn source files

REM Wind\_Loc is the location of the AeroDyn wind inflow source files

REM FAST\_LOC is the location of the FAST source files

REM ----------------------------------------------------------------------------

SET NWTC\_Lib\_Loc=C:\Users\bjonkman\Data\DesignCodes\NWTC Library\source

SET AeroDyn\_Loc=C:\Users\bjonkman\Data\DesignCodes\AeroDyn\Source

SET Wind\_Loc=C:\Users\bjonkman\Data\DesignCodes\AeroDyn\Source\InflowWind\Source

SET FAST\_Loc=C:\Users\bjonkman\Data\DesignCodes\FAST\Source

REM SET NWTC\_Lib\_Loc=D:\DATA\DesignCodes\miscellaneous\nwtc\_subs\SVNdirectory\source

REM SET AeroDyn\_Loc=D:\DATA\DesignCodes\simulators\AeroDyn\SVNdirectory\trunk\Source

REM SET Wind\_Loc=D:\DATA\DesignCodes\simulators\AeroDyn\SVNdirectory\trunk\Source\InflowWind\Source

REM SET FAST\_Loc=D:\DATA\DesignCodes\simulators\FAST\SVNdirectory\trunk\Source

REM ----------------------------------------------------------------------------

REM -------------------- LIST OF ALL SOURCE FILES ------------------------------

REM ----------------------------------------------------------------------------

SET NWTC\_Files=

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\SingPrec.f90"

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\SysIVF.f90"

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\NWTC\_IO.f90"

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\NWTC\_Num.f90"

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\NWTC\_Aero.f90"

SET NWTC\_Files=%NWTC\_Files% "%NWTC\_Lib\_Loc%\NWTC\_Library.f90"

SET Wind\_Files=

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\SharedInflowDefs.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\HHWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\FFWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\HAWCWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\FDWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\CTWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\UserWind.f90"

SET Wind\_Files=%Wind\_Files% "%Wind\_Loc%\InflowWindMod.f90"

SET AeroDyn\_Files=

SET AeroDyn\_Files=%AeroDyn\_Files% "%AeroDyn\_Loc%\SharedTypes.f90"

SET AeroDyn\_Files=%AeroDyn\_Files% "%AeroDyn\_Loc%\AeroMods.f90"

SET AeroDyn\_Files=%AeroDyn\_Files% "%AeroDyn\_Loc%\GenSubs.f90"

SET AeroDyn\_Files=%AeroDyn\_Files% "%AeroDyn\_Loc%\AeroSubs.f90"

SET AeroDyn\_Files=%AeroDyn\_Files% "%AeroDyn\_Loc%\AeroDyn.f90"

SET FAST\_Files=

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\fftpack.f"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FFTMod.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\HydroCalc.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST\_Mods.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\Noise.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST\_IO.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST\_Lin.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST2ADAMS.f90"

IF /I "%1"=="DLL" GOTO dllFiles

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\PitchCntrl\_ACH.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\UserSubs.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\UserVSCont\_KP.f90"

GOTO endFASTfiles

:dllFiles

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\BladedDLLInterface.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\UserSubs\_forBladedDLL.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\UserVSCont\_KP\_forBladedDLL.f90"

REM NOTE: UserSubs\_forBladedDLL.f90 is a copy of UserSubs.f90 with SUBROUTINES UserHSSBr() and UserYawCont() commented out

REM UserVSCont\_KP\_forBladedDLL.f90 is a copy of UserVSCont\_KP.f90 with SUBROUTINE UserVSCont() commented out

SET ROOT\_NAME=%ROOT\_NAME%\_DLL

:endFASTfiles

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\AeroCalc.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\SetVersion.f90"

SET FAST\_Files=%FAST\_Files% "%FAST\_LOC%\FAST\_Prog.f90"

:ivf

REM ----------------------------------------------------------------------------

REM ---------------- COMPILE WITH INTEL VISUAL FORTRAN -------------------------

REM ----------------------------------------------------------------------------

REM compile

ECHO.

ECHO Compiling FAST, AeroDyn, and NWTC\_Library routines to create %ROOT\_NAME%.exe:

ifort %COMPOPTS% %NWTC\_Files% %Wind\_Files% %AeroDyn\_Files% %FAST\_Files% %LINKOPTS% /out:%ROOT\_NAME%.exe

:end

REM ----------------------------------------------------------------------------

REM ------------------------- CLEAR MEMORY -------------------------------------

REM ------------- and delete all .mod and .obj files ---------------------------

REM ----------------------------------------------------------------------------

ECHO

DEL \*.mod

DEL \*.obj

SET ROOT\_NAME=

SET COPTS=

SET NWTC\_Files=

SET Wind\_Files=

SET AeroDyn\_Files=

SET FAST\_Files=

SET A2AD\_Files=

SET Fixed\_Files=

SET NWTC\_Lib\_Loc=

SET Wind\_Loc=

SET AeroDyn\_Loc=

SET A2AD\_Loc=

SET FAST\_Loc=

SET COMPOPTS=

SET LINKOPTS=

Figure 2. The "Set Compiler Internal Variables section of Compile\_FAST.bat

One way to find this command is to open the shortcut to the IVF command prompt (also called IVF Build Environment in some versions). You can usually find the shortcut at a location named something like **Start** **>** **All Programs** **>** **Intel** *CompilerName* **>** *CommandPromptName*. (Different versions of the compiler may have more submenus.) Right click on the shortcut and click “Properties.” (See Figure 3 for an example.) A window similar to Figure 4 will open.

Copy the text from the Shortcut’s “Target” field and paste it in the Compile\_FAST.bat script:

C:\Windows\SysWOW64\cmd.exe /E:ON /V:ON /K ""C:\Program Files (x86)\Intel\ComposerXE-2011\bin\ipsxe-comp-vars.bat" ia32 vs2008"

You will need to remove the call to cmd.exe and its switches, leaving you with the name of a batch file (and possibly some of its arguments):

"C:\Program Files (x86)\Intel\ComposerXE-2011\bin\ipsxe-comp-vars.bat" ia32 vs2008

If you do not want to call this batch file from Compile\_FAST.bat, you may remove the line from the file. However, you must then run Compile\_FAST.bat *only* from the compiler’s command line window. Please refer to your compiler’s documentation about using **ifort** and calling it from the command line.

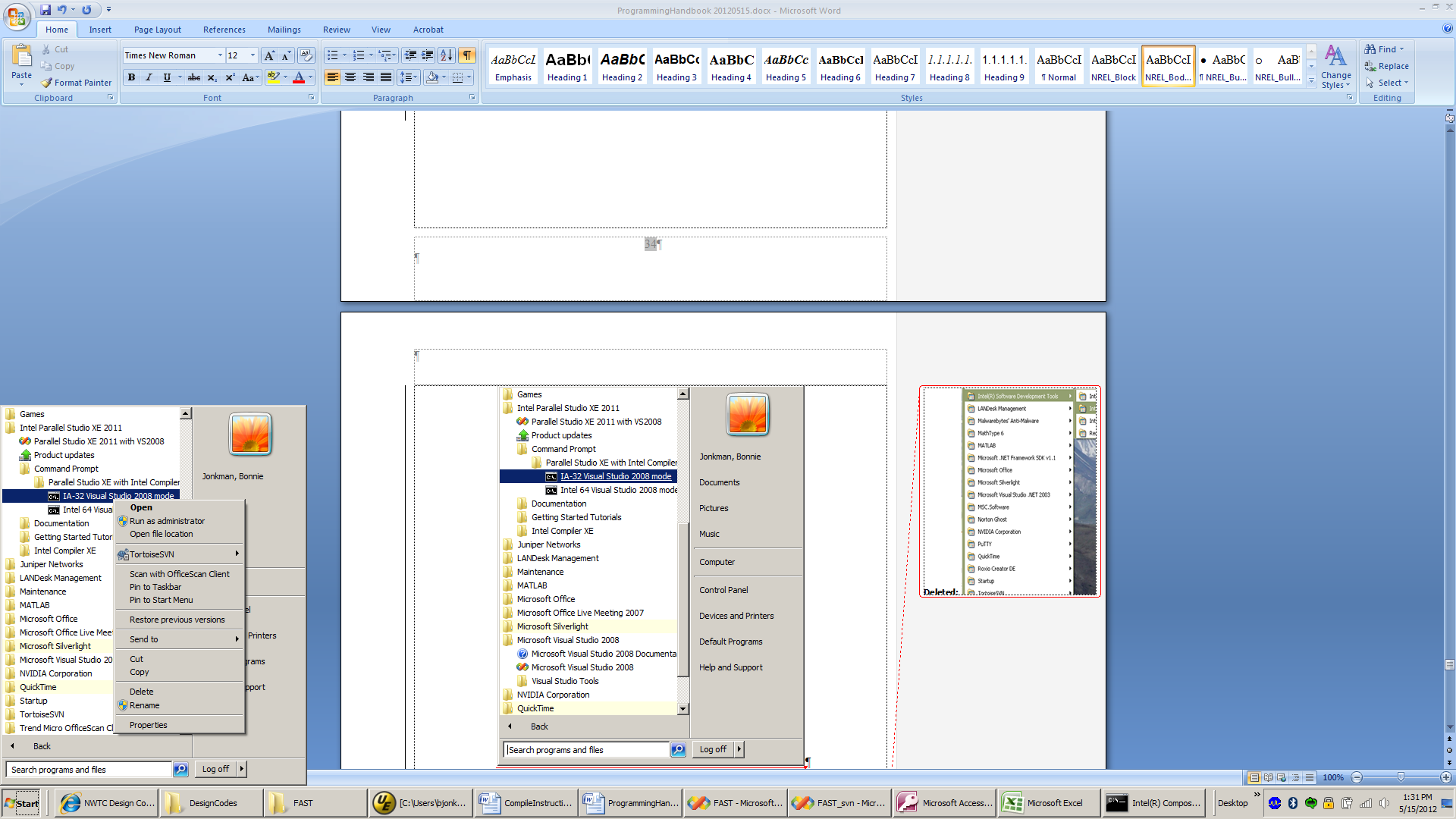


Figure 3. An example of finding the IVF command prompt shortcut

Figure . An example of finding the IVF command prompt shortcut

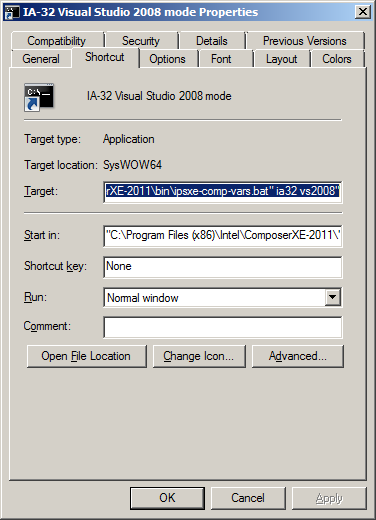


Figure 4. The properties window for an IVF command prompt shortcut

## Running the Batch File

After you have modified Compile\_FAST.bat, you can run it from the command line by typing

Compile\_FAST.bat

in the directory where the batch file is stored. Figure 5 shows the screen output after a successful build using Intel® Composer XE 2011. Notice the title of the command window after Compile\_FAST.bat has been run. The script that you call in the “Set Compiler Internal Variables” section to set the paths and environment variables for the IVF compiler also modifies the title of the window. If the title does not say anything about the compiler, please verify that you have modified Compile\_FAST.bat correctly.

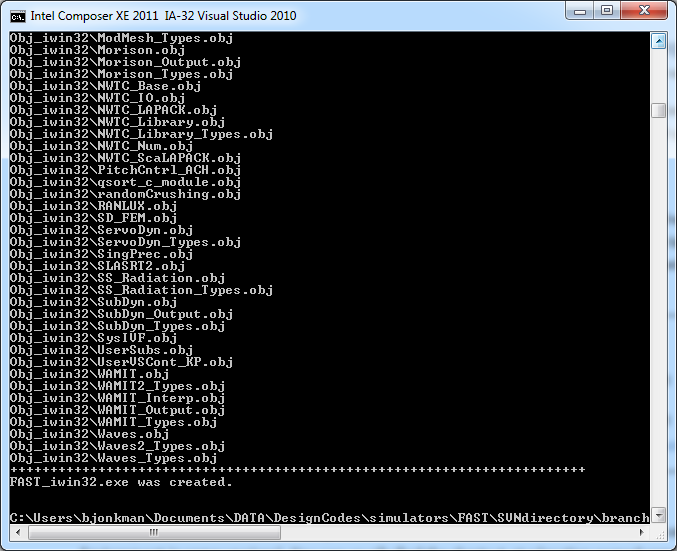


Figure 5. Screen output from Compile\_FAST.bat

# Compiling with gfortran for Windows

The makefile in the FAST\Compiling folder can be used to compile FAST v8 using gfortran.

gfortran (gcc) for Windows will require you to download the LAPACK and BLAS binary files, which you can get here: <http://icl.cs.utk.edu/lapack-for-windows/index.html>. For 32-bit Windows, you will need these files[[1]](#footnote-1):

* <http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win32/liblapack.dll>
* <http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win32/liblapack.lib>
* <http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win32/libblas.dll>
* <http://icl.cs.utk.edu/lapack-for-windows/libraries/VisualStudio/3.5.0/Dynamic-MINGW/Win32/libblas.lib>

When you use gfortran for Windows, you will need to make sure these .dll files are on your Windows PATH so that the executables will run. The .lib files are required for linking.

To use the LAPACK and BLAS libraries in gfortran, use linking options –llapack and –lblas. (if you are having trouble, this web site may also provide some useful information: <http://www.math.utah.edu/software/lapack.html>)

# Compiling and Linking Options

**The options required to compile and link FAST v8.08.00c-bjj are already set in the compiling tools distributed with the code.** However, it is sometimes useful to modify the options for debugging or other purposes.

Table 1 lists several useful compiling options. The first column of the table indicates which options are required to compile FAST.

We also require some additional linking options:

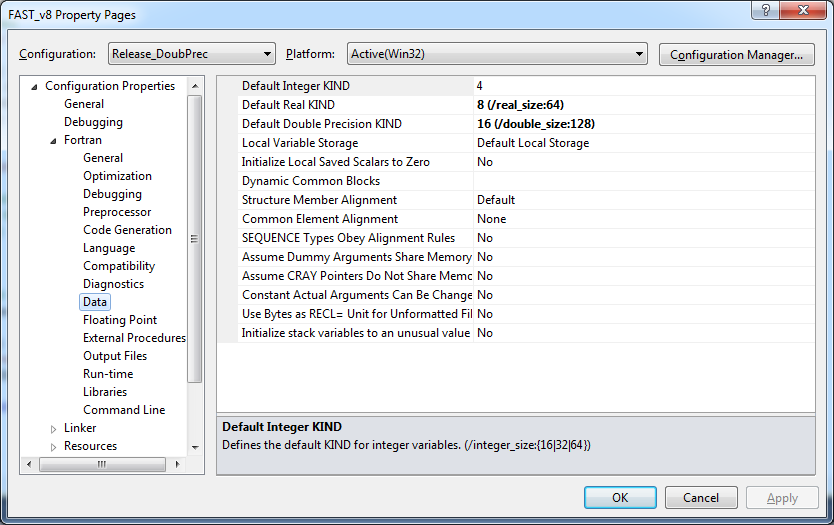
* You must link with the MAP library. On Windows, this means adding MAP\_Win32.lib (or MAP\_x64.lib) to the linking command. On non-Windows machines, this means adding MAP\_{OS name}.so to the linking command.
* SubDyn places some large matrices on the stack, and we have found that on Win32 systems, we need to increase the stack reserve size. The command option is:  
  /STACK:999999999 (IVF)  
  -Wl,--stack=999999999 (gfortran)
* Also on some large models (e.g., OC4 Jacket), we can exceed the 2-GB Windows memory limit for 32-bit processes. We can extend this limit when running the application on 64-bit computers by adding the option to Enable Large Addresses.  
  /LARGEADDRESSAWARE (IVF)  
  -Wl,--large-address-aware (gfortran)

If you compile a 64-bit version of FAST, you can omit the /STACK and /LARGEADDRESSAWARE options.

## Double Precision

To compile FAST in double precision:

* Use the DoubPrec.f90 file instead of SingPrec.f90 from the NWTC Subroutine Library.
* If you are using the FFTpack routines (i.e., HydroDyn), you will also need to set the default Real and Double KINDs to 8 and 16 bytes, respectively. (Default REAL must be the same as ReKi and Default Double must be the same as DbKi.)



## LAPACK Libraries

FAST v8 uses some LAPACK routines (<http://www.netlib.org/lapack/index.html>). We have made the decision to link with prebuilt libraries, which should include highly optimized versions of the Basic Linear Algebra Subprograms (BLAS).

These prebuilt libraries typically come installed on Linux and Mac operating systems. Use linking options –llapack and –lblas for gfortran.

If you are using the Intel compiler, you can use the Intel® Math Kernel Library. To activate MKL, you will need to set *Project* -> {project name} *Properties* -> *Configuration* *Properties* -> *Fortran* -> *Libraries* -> *Use Intel Math Kernel Library* to **Sequential (/Qmkl:sequential)**.

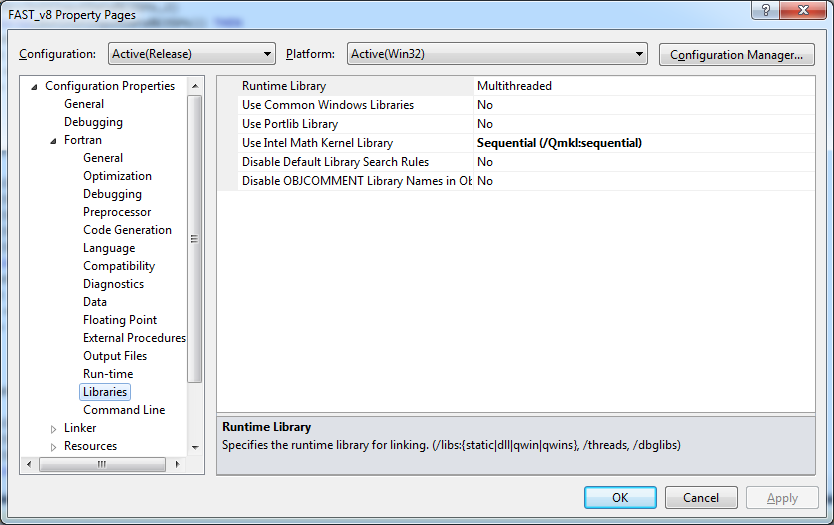


Figure 6. Setting Visual Studio to use LAPACK

With this option set, you can compile and use any routine in the LAPACK libraries.

If you have an older version of the Intel compiler, you may not have access to MKL. In that case, please see the instructions at this web site: <http://icl.cs.utk.edu/lapack-for-windows/lapack/> for help in installing LAPACK and BLAS libraries for Release and Debug modes.

### LAPACK in Double Precision

To facilitate compiling in double precision, we have created some wrapper routines in for the LAPACK and ScaLAPACK routines used in FAST and its modules.

*The NWTC wrapper routines for LAPACK have been written assuming that they are calling prebuilt libraries*. If you choose to compile with the LAPACK source files instead of using the prebuilt libraries (not recommended), you must make sure that default Real and Double KINDs for the LAPACK routines are 4 and 8 bytes, respectively (*even when compiling FAST in double precision*).

Table 1: Fortran Compiling Options

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Used in FAST?** | **Intel Fortran**  **(Windows)** | **Intel Fortran**  **(Linux and Mac)** | **GNU Fortran** |
| Data options |  |  |  |  |
| Define all default real (and complex) variables as 4 bytes long | Recommended  *(compiler default)* | /real-size:32 | -real-size 32 | *(set by default)* |
| Define all default real (and complex) variables as 8 bytes long |  | /real-size:64 | -real-size 64 | -fdefault-real-8 |
| Optimization options |  |  |  |  |
| Disable all optimizations (debug mode) | Recommended for Debug | /Od | -O0 | -O0 |
| Enable optimizations for speed (release mode) | Recommended for Release | /O2 | -O2 | -O2 |
| Enable higher optimizations (may set other options; may not be appropriate for all codes) |  | /O3 | -O3 | -O3 |
| External libraries |  |  |  |  |
| Use optimized LAPACK routines | **Required** | /Qmkl:sequential | -mkl=sequential | -llapack -lblas[[2]](#footnote-2) |
| Debugging options |  |  |  |  |
| Provide source file traceback information when a severe error occurs at run time |  | /traceback | -traceback | -fbacktrace |
| Check array subscripts |  | /check:bounds | -check bounds | -fcheck=bounds |
| Fortran Dialect Options |  |  |  |  |
| Produce warning/error for non-standard Fortran 2003 code | Recommended | /stand:f03 | -std03 | -std=f2003 |
| Allow free-format code to exceed 132 columns | **Required** | *(allowed by default)* | | ‑ffree‑line‑length‑none |
| Other |  |  |  |  |
| Display compiler version information |  | /logo | -logo  -V | -v  --version |
| Preprocess source files before compilation | **Required** | /fpp | -fpp | -x f95-cpp-input |
| Create 32-bit code |  | *(use appropriate Visual Studio configuration or call appropriate script prior to using compiler from command line)* | | -m32 |
| Create 64-bit code |  | -m64 |

# Compiling on Linux or Mac

Some of the biggest challenges to compiling on non-Windows® systems come from the MAP dynamically linked library and the FAST Registry.

We have included the source code for the FAST Registry in the Source\dependencies\Registry folder. That folder also contains a makefile that will generate the FAST executable using gcc. It has been tested and run on a few Linux and Mac systems.

The MAP library is a little more difficult. Because of some of the third-party libraries compiled in MAP, it is difficult to compile for different systems. If you do not need to actually call MAP, you can compile an empty MAP dll with source files and a makefile in the FAST archive’s Source\dependencies\Blank\_MAP\_DLL folder. We are working to update MAP so that it is easier to compile on different systems.

# Feedback

If you have questions or suggestions for improvements, please use our forums: <https://wind.nrel.gov/forum/wind/>

1. At the time of this writing, the web site had typos in the links for liblapack.dll and liblapack.lib; use the links listed here. [↑](#footnote-ref-1)
2. These are actually linking options in gfortran. [↑](#footnote-ref-2)