# ReadMe File for FAST v8.03.02b-bjj

Bonnie Jonkman, Jason Jonkman  
National Renewable Energy Laboratory  
October 3, 2013

## Introduction

This document is designed to guide you through some of the changes that the FAST wind turbine computer-aided engineering (CAE) tool is undergoing. FAST v8.03.02b-bjj is the first public release of FAST under the [new modularization framework](http://wind.nrel.gov/designcodes/simulators/developers/) developed at NREL. The architecture of FAST v8 is entirely different from FAST v7.02.00d-bjj. These differences are highlighted in Figure 1.



Figure : Architectural comparison of FAST 7 and FAST 8[[1]](#footnote-1)

The modules of FAST (AeroDyn, HydroDyn, etc.) correspond to different physical domains of the coupled aero-hydro-servo-elastic solution, most of which are separated by spatial boundaries. Figure 2 shows the control volumes associated with each module for fixed-bottom offshore wind turbines. For land-based wind turbines, the HydroDyn hydrodynamics module would not be used and the SubDyn multi-member substructure structural-dynamics module is optional. Figure 3 shows the control volumes associated with each module for floating offshore wind turbines.



Figure : FAST control volumes for fixed-bottom systems



Figure : FAST control volumes for floating systems

Many features of [FAST v7.02.00d-bjj](http://wind.nrel.gov/designcodes/simulators/fast) have not yet been added to FAST v8, so we will continue to support both versions of the software (FAST v7 and FAST v8) until FAST v8 is deemed a suitable replacement. Table 1 summarizes the different features available in each version.

Table : Comparison of features between FAST v7 and v8

|  |  |  |
| --- | --- | --- |
| **Aerodynamics (AeroDyn and InflowWind)** |  |  |
| **FAST Features** | **v7.02** | **v8.03** |
| • Quasi-steady or dynamic wake |  |  |
| • Steady or unsteady airfoil aerodynamics |  |  |
| • Tower shadow for downwind rotors |  |  |
| • Tower influence for upwind rotors |  |  |
| • Tower drag loading |  |  |
| • Tail-fin aerodynamic loading |  |  |
| • "Hub-height", TurbSim, and GH Bladed wind file formats |  |  |
| • Other wind formats |  |  |
| • Aeroacoustics (noise) |  |  |
| **Hydrodynamics (HydroDyn)** |  |  |
| **FAST Features** | **v7.02** | **v8.03** |
| • Linear regular or irregular waves |  |  |
| • White-noise waves |  |  |
| • Wave stretching |  |  |
| • Externally generated wave data |  |  |
| • Sea current |  |  |
| • Morison's equation for central member |  |  |
| • Morison's equation for multiple intersecting members |  |  |
| • Static buoyancy and dynamic pressure on members |  |  |
| • Support for inclined and tapered members |  |  |
| • Support for flooded and ballasted members |  |  |
| • Support for marine growth |  |  |
| • First-order potential flow (from WAMIT) |  |  |
| • Radiation "memory effect" captured through time-domain convolution |  |  |
| • Radiation "memory effect" captured through linear state-space form |  |  |
| **Control and Electrical System (Servo) Dynamics (ServoDyn)** |  |  |
| **FAST Features** | **v7.02** | **v8.03** |
| • Blade-pitch control |  |  |
| • Override pitch maneuvers |  |  |
| • Generator models |  |  |
| • Torque control |  |  |
| • High-speed shaft brake |  |  |
| • Nacelle-yaw control |  |  |
| • Override yaw maneuvers |  |  |
| • Blade-tip brakes |  |  |
| • GH Bladed DLL interface[[2]](#footnote-2) |  |  |
| • Simulink interface |  |  |
| • LabVIEW interface |  |  |
| **Structural Dynamics (ElastoDyn, SubDyn, and MAP)** |  |  |
| **FAST Features** | **v7.02** | **v8.03** |
| • Blade-bending DOFs |  |  |
| • Rotor teeter DOF |  |  |
| • Generator azimuth and drivetrain torsion DOFs |  |  |
| • Nacelle-yaw DOF |  |  |
| • Tower-bending DOFs |  |  |
| • Rigid-body platform DOFs |  |  |
| • Furling DOFs |  |  |
| • Fixed-bottom multi-member substructure DOFs: |  |  |
| − Solved with linear frame finite-element or Craig-Bampton reduction |  |  |
| • Gravitational loading |  |  |
| • Gearbox friction |  |  |
| • System of independent mooring lines solved quasi-statically |  |  |
| • System of multi-segmented mooring lines solved quasi-statically |  |  |
| • Earthquake excitation |  |  |
| **General** |  |  |
| **FAST Features** | **v7.02** | **v8.03** |
| • Time marching |  |  |
| • Operating-point determination |  |  |
| • Linearization |  |  |
| • FAST-to-ADAMS preprocessor |  |  |
| • Follows the new FAST modularization framework |  |  |
| • Structural and control routines separated from driver code |  |  |
| • Independent time steps between modules | [[3]](#footnote-3) |  |
| • Independent spatial discretization between modules |  |  |
| • Multiple integration options |  |  |
| • Loose coupling with predictor-corrector across modules | [[4]](#footnote-4) |  |
| • Both 32-bit and 64-bit applications available |  |  |
| • Supports both Windows and Linux operating systems |  |  |
| • Optimized for efficiency |  |  |
| • Supports mixed Fortran/C |  |  |
| • Compiles with gfortran[[5]](#footnote-5) |  |  |

## Major changes in FAST v8.03.02b-bjj

Tasks completed to develop FAST v8 included:

* Converted FAST and its various modules (including AeroDyn and HydroDyn) into the [new modularization framework](http://wind.nrel.gov/designcodes/simulators/developers/) (splitting out the controls and electrical drive dynamics into a new ServoDyn module and structural dynamics into a new ElastoDyn module),
* Implemented a new driver program (glue code) supporting loose coupling of the modules,
* Developed mesh-to-mesh mapping schemes between module-independent discretizations of the spatial boundaries between modules,
* Coupled in the recently developed SubDyn module for multi-member substructure structural dynamics and MAP module for multi-segmented mooring quasi-statics, and
* Included a series of models using the NREL 5-MW Baseline wind turbine in the CertTest, including offshore configurations.

The driver program (glue code) couples the modules together; it controls the overall simulation progress and maps module outputs to inputs. We use the name “FAST” both for the driver program (glue code) and overall coupled code.

FAST v8.03.02b-bjj is compiled with the following components:

|  |  |  |
| --- | --- | --- |
| Component | | Version |
| Modules |  | |
| ElastoDyn | v1.01.02b-bjj | |
| ServoDyn | v1.01.01b-bjj | |
| AeroDyn | v14.02.00c-mlb | |
| InflowWind | v2.00.00b-adp | |
| HydroDyn | v2.00.10a-gjh | |
| SubDyn | v0.04.00a-rrd | |
| MAP | v0.87.06a-mdm | |
| Other Components |  | |
| NWTC Subroutine Library | v2.03.00b-bjj | |
| FAST Registry[[6]](#footnote-6) | v2.01.00 | |

FAST and each of its modules, except InflowWind, have their own input files; see Figure 4 and the section “Converting from FAST v7.x to FAST v8.03.x” of this document.

## Limitations

We are working to add the features of FAST v7.02.00d-bjj into v8. Table 1 shows a comparison of features, including the limitations of FAST v8.03.x.

At this time, while ElastoDyn has been coupled to HydroDyn for floating offshore wind turbines and ElastoDyn has been coupled to SubDyn for fixed-bottom wind turbines, SubDyn has not yet been coupled to HydroDyn. So, it is not possible with this release to model hydrodynamic loads on fixed-bottom offshore wind turbines.

Please note that FAST v8.03.x runs significantly slower than FAST v7.02.x. This is especially true of any model that uses the coupling between ElastoDyn and HydroDyn or SubDyn. We have put our effort into getting the framework to work and will address computational efficiency later. We expect great improvements in efficiency as development continues.

*While we are very excited about this release and the new capability it brings to FAST, please be aware that this is the initial release. As with anything new, we encourage users to take appropriate precautions. We have tested many features, but not all. Please interpret the results carefully and report back any confirmed errors to the NREL developer. Further development, verification, and validation work is ongoing at NREL. Check back regularly to obtain the latest version of the code.*

## Future Work

All future developments of FAST will follow the framework.

* Items from the “Limitations” section will be addressed
* Couple SubDyn and HydroDyn to enable the modeling of fixed-bottom offshore wind turbines
* Introduce the new BeamDyn module for nonlinear finite-element modeling of blade dynamics
* Upgrade the loose coupling algorithm of the glue code to allow each module to have its own time step
* ElastoDyn will have a separate discretization scheme for the blades and tower (currently ElastoDyn uses AeroDyn’s blade discretization and AeroDyn uses ElastoDyn’s tower discretization)
* Optimize the code, particularly ElastoDyn, so that it runs faster
* Introduce tight coupling
* Introduce operating-point determination and linearization across the coupled aero-hydro-servo-elastic solution
* And much, much, more…

## Converting from FAST v7.x to FAST v8.03.x

We have created template input files for FAST v8.03.02b-bjj, ElastoDyn v1.00.02b-bjj, and ServoDyn v1.01.01b-bjj. These template files can be found in the Matlab Simulation Toolbox that is now included in the FAST archive: Utilities\SimulationToolbox\ConvertFASTVersions\TemplateFiles\v8.03.x

### Summary of Changes to Inputs

* The primary FAST input file has been converted to primary input files for FAST, ElastoDyn, and ServoDyn and some of the inputs have been reordered.
* The FAST Platform file has been eliminated, with some of the inputs now part of the ElastoDyn primary input file and some of the inputs now part of HydroDyn’s and MAP’s input files.
* All of the inputs formerly labeled “[CURRENTLY IGNORED]” have been removed.
* Switches for ADAMS preprocessing and linearization have been removed.
* Noise has been removed.
* PtfmLdMod has been converted to CompUsrPtfmLd.
* TwrLdMod has been converted to CompUserTwrLd.
* The tip-brake inputs have been removed.
* PtfmCM is now PtfmCMzt, with PtfmCMzt = -PtfmCM.
* Corresponding inputs PtfmCMxt and PtfmCMyt have been added.
* PtfmRef is now PtfmRefzt, with PtfmRefzt = -PtfmRef.
* TwrRBHt and TwrDraft have been replaced with TowerBsHt, with   
  TowerBsHt = TwrRBHt – TwrDraft.
* The output decimation factor (DecFact) has been converted to DT\_out (DT\_out = DT\*DecFact).
* The yaw and pitch maneuvers no longer specify end times for the maneuvers. Instead they specify a rate for the maneuver.
* The GBRevers variable has been removed; input GBRatio must now be specified as a negative number if GBRevers was previously set to TRUE.
* ElastoDyn’s blade input properties table no longer specifies AeroCent. Instead, it specifies the location of the pitch axis, PitchAxis, which is calculated as PitchAxis = 0.5 – AeroCent by the MATLAB conversion script; the aerodynamic center will become part of AeroDyn in a future release.
* The OutList variables have been divided among the various FAST modules, and several outputs are no longer valid.
* The GH Bladed Interface is now a standard option in ServoDyn, without requiring a recompile.
* Tower drag loading has been added to AeroDyn v14.02.00c-mlb with a new corresponding flag in the AeroDyn input file.
* The glue code allows options for AbortErrLevel, number of corrections in the predictor-corrector algorithm, and extrapolation/interpolation order of module inputs to be used for time advancement.
* For the ElastoDyn coupling to HydroDyn or SubDyn, FAST also has two inputs controlling the implicit solve (via Jacobian computed with finite differences). UJacSclFact is used to scale the loads from HydroDyn or SubDyn i so that they are roughly the same of magnitude as the accelerations from ElastoDyn; DT\_UJac determines how often the Jacobian needs to be updated (if the platform reference point doesn't rotate much, DT\_UJac can be set to a value larger than TMax).

### Matlab Conversion Script

Because the changes to the input files are significant, we have created Matlab scripts to automatically convert FAST v7.x input files to FAST 8.03.02b-bjj input files. The files you will need are included in the Simulation Toolbox, located in this directory of the FAST archive: Utilities\SimulationToolbox\ConvertFASTVersions

We recommend that you add the Simulation Toolbox to your Matlab path so that you can access all of the routines defined in it. For example:

FASTSimulationToolbox = 'C:\Users\bjonkman\FAST\UtilityCodes\SimulationToolbox';

addpath( genpath( FASTSimulationToolbox ) );

An example showing how we converted the NREL CertTest input files for use with FAST v8.03.02a-bjj is included in the FAST archive: CertTest\ConvertFiles.m. You can use this script as a basis for helping to convert your own input files; however, we *strongly* recommend that you make copies of all your input files before running any scripts to convert them.

You will need to provide the conversion routine (ConvertFAST7to8) with the name of the old FAST (v7.x) primary input file and the directory where the new input files should be placed. *The new directory should not be the directory where the old files are located!*

If your input file has pitch or yaw maneuvers, you may also provide the routine with the new rates (instead of the end times previously used). We have also provided a Matlab routine (CalculateYawAndPitchRates) that will calculate these rates, but you must provide the routine the name of the FAST output file that contains the previous results of the Pitch and/or Yaw channels.

If your input file was used with the custom interface to GH Bladed DLL controllers, you should also set the optional input parameter, usedBladedDLL, so that your input switches that previously called the DLL are now set to 5, the new switch for User-Defined Control from Bladed DLL.

Please note that the hydrodynamic- and mooring system-related inputs are not automatically converted.

### Model Time Steps

Please note that due to some of the changes in the coupling scheme of FAST v8, you may need to change the time steps in your existing models. (Currently, we use lock-step coupling, so all modules must have the same time step as the FAST driver.) For example, we have noticed that the NREL 5-MW Baseline model must use a smaller time step.

## Compiling

If you want to compile the code, please see the compiling folder in the FAST archive.

Unlike previous FAST distributions, all of the source code you need to compile the project is contained in the archive’s Source directory, allowing us to distribute a Microsoft Visual Studio project file (and solution) that contains links to all of the files needed to compile FAST v8.03.02b-bjj.

The compiling folder also contains a Windows® batch script*[[7]](#footnote-7)* that can be run from your Intel Fortran Command Prompt Window, with very little (if any) modification. Both the Visual Studio project and batch file run the FAST Registry and are set up to compile and link with all of the appropriate settings. (Note that there are several specialized compiling/linking options and that MAP is distributed as a dynamic-link library.)

The text files (.txt) in the source folders are input files for the FAST Registry. These files are used to generate the \*\_Types.f90 files for the component modules. The Compiling\Compile\_FAST.bat file and Compiling\VisualStudio\RunRegistry.bat files will run the Registry if necessary.

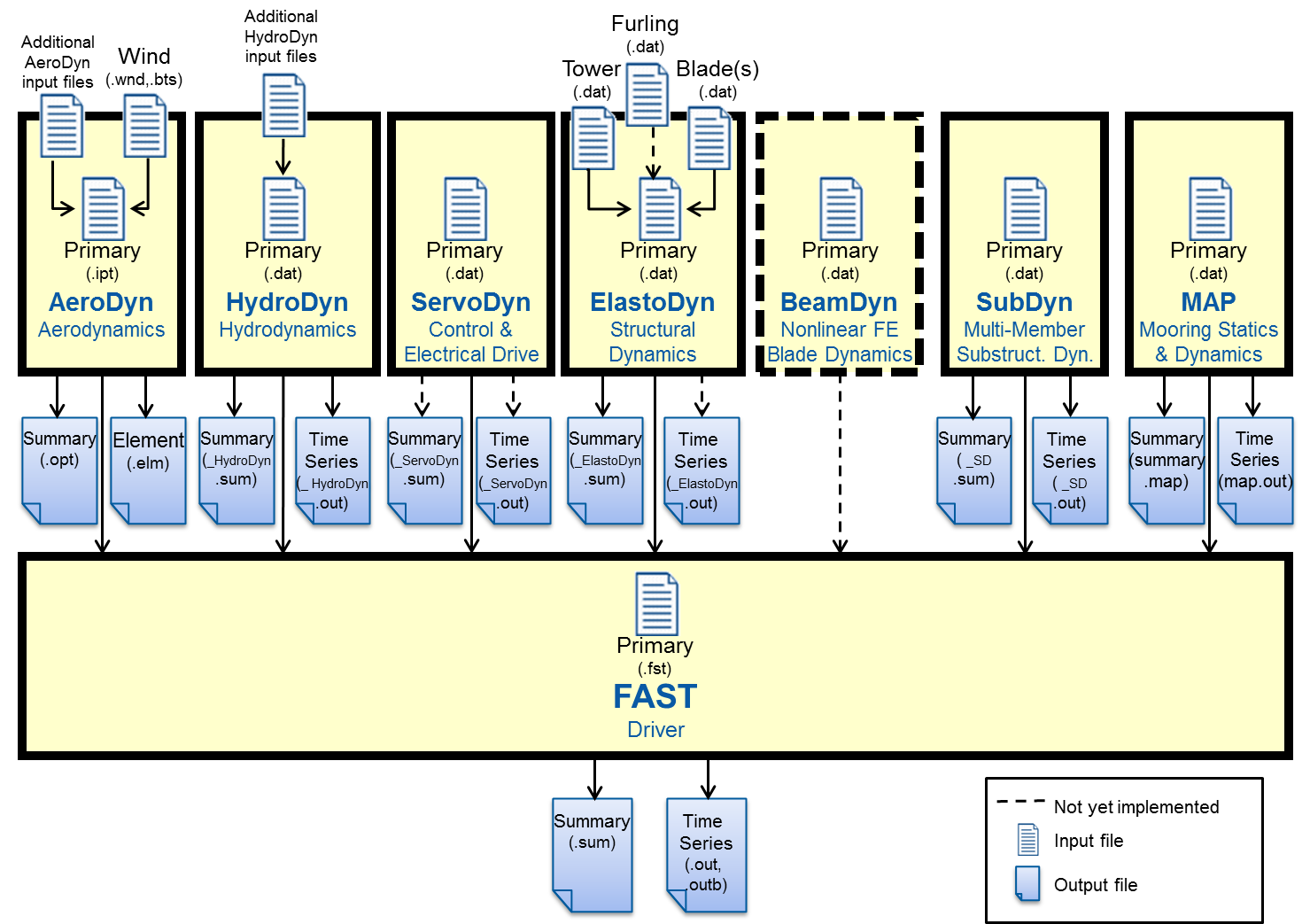


Figure : Summary of Input and Output Files for FAST v8.03.02b-bjj

1. The BeamDyn nonlinear finite-element blade-dynamics module is under development and will be released in 2014. [↑](#footnote-ref-1)
2. This option is a custom feature in FAST v7.02.x, requiring a separate executable. In FAST v8.03, it is part of the standard distribution. [↑](#footnote-ref-2)
3. These steps must be integer multiples of the structural time step. [↑](#footnote-ref-3)
4. FAST v7.02.x is limited to one correction step and this correction step only applies to some modules. [↑](#footnote-ref-4)
5. Most of FAST v8.03.x source code can be compiled using gfortran, however SubDyn has some dependencies on Intel Fortran that we are working to replace. [↑](#footnote-ref-5)
6. The FAST Registry v2.01.00 reads input files from each module to auto-generate the \*\_Types.f90 files. [↑](#footnote-ref-6)
7. Note that the included *makefile* does not currently work due to some of the code’s dependence on Intel Fortran. However, it does contain an accurate list of source files. We are working to get the *makefile* working with gfortran in the near future. [↑](#footnote-ref-7)