Overview of C++test Support for Wind River Tornado

This topic provides information about using C/C++test to test source code that is designed to be compiled/built using Wind River Tornado compilers and/or written with the aid of the Wind River Tornado IDE. It covers the C/C++test features that specifically support Tornado, as well as the Tornado features that you should understand in order to perform testing with C/C++test. For additional details on general C++test functionality, refer to the other parts of the this user guide.

Support Overview

Both C/C++test standalone and the C/C++test Eclipse plugin allow C/C++test to be used with Tornado—no special integration is required, because C/C++test's support for Tornado does not involve integrating with the Tornado development environment. Instead, C/C++test is preconfigured to support Tornado's set of compilers, allow C/C++test projects to use these compilers, and allow complete Tornado projects to be imported into the C/C++test workspace.

Supported Tornado Versions and Compilers

C/C++test supports Tornado 2.0 and 2.2. Both versions come with their own compilers:

- Tornado-2.0 includes a clone of GNU EGCS: egcs-2.90.
- Tornado-2.2 includes two compilers: a GNU GCC clone (gcc-2.96) and Wind River's own DIAB compiler (dcc Rel 5.0).

This table lists Tornado-shipped compilers and details their C/C++test family and configuration names*:

Tornado Version	Shipped Compilers	C++test Family	C++test Configurations
2.0	egcs-2.90	Wind River EGCS 2.9	wregcs_2_9
2.2	gcc-2.96, DIAB-5.0**	Wind River GCC 2.9, Wind River Diab 5.0**	wrgcc_2_9 , diab_5_0

*The configuration name is the name of a subdirectory under the C/C++test compilers' main configuration directory, which is where the compilers' configuration data are stored. By default, this is at `C++test_install_dir/engine/etc/compilers/config_name`.

**Or higher DIAB version starting from 5.0.1.

The VxWorks versions shipped:

- Tornado-2.0: VxWorks-5.4 (C++test support is deprecated)
- Tornado-2.2: VxWorks-5.5 (C++test support is deprecated)

Prerequisites

For C/C++test itself, the standard rules apply. If you have the standalone C/C++test Eclipse product installed, then no extensions are needed (CDT is already included). If you have an Eclipse plugin, then you must also install the CDT (C++ Development Tools) extension into Eclipse. See Installation for details.

To allow testing with Tornado compilers, you must have the following special Tornado environment variables set *prior to launching C/C++test* (we assume that you either have the Tornado development environment installed properly or you have one or more Tornado compilers installed by other means):

- **WIND_BASE**: Tornado installation directory. For example: C:\Tornado.
- **WIND_HOST_TYPE**: Host type of the machine on which Tornado is installed for the GNU toolchain. Currently, only `x86-win32` is supported.
- **DIABLIB - The DIAB** installation directory. In most cases, this is `%WIND_BASE%\host\diab`. This variable is necessary only if you have Tornado-2.2 and you want to test with the DIAB compiler.
- **DIAB_HOST_TYPE** - The host type of the machine on which Tornado is installed for the DIAB toolchain. Currently, only WIN32 is supported (only when you want to test with DIAB compiler).

We also recommended putting the Tornado executables on the PATH variable. For example,

```
"set PATH=%WIND_BASE%\host\%WIND_HOST_TYPE%\bin;%PATH%"
"set PATH=%DIABLIB%\WIN32\bin;%PATH%"
```

The Tornado-shipped batch script (`%WIND_BASE%\host\%WIND_HOST_TYPE%\bin\torVars.bat`) performs all recommended environment initialization. When you run C/C++test from a POSIX-like shell, you can create a shell script based on `torVars.bat`.

For static analysis, setting the environment is sufficient. For unit testing, the following tools must also be present (see Executing Test Objects):

- Tornado Registry (wtxregd)
- Target Server (tgtsvr)
- Tornado Shell (windsh)

Known Limitations

Limitations of C/C++test:

- Only the Eclipse-based C/C++test versions (standalone and plugin) support testing for Tornado.
- Only the Tornado for Windows platform is officially supported.
- Support for VxWorks versions 5.4 and 5.5 is deprecated--testing for later versions is supported, but not in the context of the Wind River Tornado environment.
- Testing on platforms other than VxSim requires adjustment to the mangling schema - see Mangling Workaround
- There may be problems with catching exceptions when using the DIAB compiler.

Limitations of Tornado tools:

- The wregcs-2.9 compiler from Tornado-2.0 doesn't support namespaces and using-directives (except synthetic "using namespace std;").
- The wregcs-2.9 preprocessor doesn't accept include paths containing spaces. To avoid this issue, don't install C/C++test to such locations (e.g., "Program Files").
- There are many Tornado old make-3.74 limitations (see Accounting for Make Varieties) that prevent it from parsing much of today's normal makefiles. Consequently, the C/C++test Runtime library's makefiles (see Understanding and Building the Runtime Library) must be specially adjusted to it. You may experience problems with building your non-Tornado projects when this make is put on the PATH; as a result, it's best to set the environment for Tornado only when you mean to use Tornado tools only.

Limitations of VxWorks-5.4 & VxWorks-5.5:

- Standard libraries do not support "long long" or the "long double" type; moreover, they do not support any "strtoll", "strtold" routines or "%lld", "%llu" format strings of "printf"-family routines.
- The default build of VxWorks-5.5 simulator (%WIND_BASE%\target\config\simpc\vxWorks.exe; Tornado-2.2) doesn't contain enough C++ features to satisfy C/C++test's C++ instrumentation needs in its default mode. Thus, to test C++ code, you either need to build your own VxSim image with scaled-up C++ support (we recommend using all C++ features except complex numbers; at minimum) or you need to add the -DCPPTEST_SPECIAL_STD_EXCEPTIONS_HANDLING_ENABLED=0 definition to the project compilation flags. This issue does not affect VxWorks-5.4 (Tornado-2.0).

Mangling Workaround

Tornado's GNU toolchain uses different mangling schemes for the simulator (VxSim) and for other platforms. For VxSim, the scheme is adjusted to be Microsoft/Windows/Cygwin-compatible with additional underscores prepended to C-symbols and other differences in complex C++ mangling.

In C++test nomenclature, VxSim mangling schemes are called gcc-cygwin/g++2-cygwin respectively for C/C++, while names for all other platforms are gcc /g++2. C/C++test is prepared to test initially for VxSim and currently unable to adjust its mangling scheme automatically. As a result, you need to make this adjustment manually for other platforms.

There are two ways to adjust this mangling scheme manually:

- Providing the symmatcher.manglingSchema Advanced Project Option (in the **Other Settings> Advanced Options** area of the C/C++test project properties panel) set to gcc for C-only projects or to g++2 for C++-only projects. This solution does not apply to mixed C/C++ projects.
- Creating a Custom Compiler by duplicating the wregcs-2.9/wrgcc-2.9 configuration, adjusting its schemes, and selecting it for the project.

Here are the details on how to use the second method listed above:

1. Choose **File> New> Other**.
2. Select **C++test> Custom Compiler**, then click **Next.**
3. Select **Add custom compiler**, then click **Next**.
4. Specify the custom compiler as follows:
    a. Enter a meaningful name for the compiler.
    b. Select either **Wind River EGCS 2.9** or **Wind River GCC 2.9** under **compiler family**.
    c. Provide the correct compiler/linker executables.
5. Make any desired adjustments to the compiler's identifier (e.g. if the configuration is to be shared), then click **Next.**
6. In the last 3 edit boxes, mark the location where your compiler configuration files are to be generated, then click **Finish.**
7. Using an external file system navigator, open the marked location.
8. Edit the c.psrc file and append `symmatcher.manglingSchema gcc`
9. Edit the cpp.psrc file and append `symmatcher.manglingSchema g++2`
10. Save both files.
11. To prompt C/C++test to pick the changes, restart it.
    - Alternatively, if you adjusted the C++test Preferences's **Configurations> Custom directories> Custom compilers** location, you can disable it, click **Apply**, re-enable it, then click **OK**.
12. Select your new compiler configuration for any project in the **Build Settings> Compiler settings> Family** area of the C/C++test project properties panel.