Importing Projects with IAR Embedded Workbench Support

Creating a C++test Project from an EWARM Project

1. Choose **File> New> Project** and select **Import IAR Embedded Workbench projects**.
2. Click **Next** and type (or browse to) the path to appropriate project/workspace file or directory.
3. Click the **Refresh** button and select projects to import.
4. Click **Next** and set the import destination, project contents and build configuration to use.
5. Click **Finish**.

Creating a C++test Project from Other EW Project Types

C++test's built-in GUI-based EW project importer only supports EWARM projects, and only for specific EWARM versions (see Support Overview). Importing EW projects for other toolchain versions or target architectures must be done with the `cpptesttrace` utility. The `cpptesttrace` utility scans the command lines for processes spawned by the `iarbuild` utility.

The importing process consists of two stages: 1) Generating the Build Data File (.bdf) and 2) importing this file in C++test.

Generating a .bdf from an EW Project

1. For runtime testing, increase "Stack/Heap" in EW project options to at least 2048(800h)/ 1280(500h) respectively to allow for testing overheads. For EW430 projects that use the 430X core and medium or large data model, the values should be at least 4096(1000h) for stack and 2048(800h)/4096(1000h) for the heaps.
2. Start console.
3. Set the compiler toolchain on PATH (as for C++test; 'cspybat' on PATH & EW_DIR envar aren't needed).
4. Set the C++test install directory on PATH.
5. CD to the project's directory.
6. Run the following command:

```
cpptesttrace --cpptesttraceProjectName=<prj_name> --
cpptesttraceResponseFileOption=-f --
cpptesttraceOutputFile=<prj_dir>\<prj_name>.bdf
iarbuild <prj_name>.ewp -build <EW_project_config_name>
```

Use an EW project configuration name (e.g., `Debug`, `Release`, `DebugRunFromFlash`, etc.) for the `-build` argument.

Specify the absolute path to the `--cpptesttraceOutputFile` option. This is because several .bdf files are likely to spread through working directories spawned by `iarbuild` toolchain commands.

You may need to place quotation marks around the options containing spaces/braces.

In most circumstances, you should name your C++test project after the .ewp file name when using the `--cpptesttraceProjectName=<prj_name>` option.

By default, the trace feature is configured to capture each toolchain executable name recognized by C++test through built-in compiler configurations. If the process' command-lines aren't scanned, specify the compiler and linker executable names by adding a `--cpptesttraceTraceCommand` option to the `cpptesttrace` command.

EW430 Example:

```
--cpptesttraceTraceCommand=icc430\.exe$|xlink\.exe$
```

STM8 Example:

```
--cpptesttraceTraceCommand=iccstm8\.exe$|ilinkstm8\.exe$
```

The `cpptesttrace` command always appends its output file, so you should remove the previous .bdf file before re-scanning the project.

You may also use `-log all` after the `-build` flag to show detailed build progress information. The `-log all` option is an `iarbuild` option.

Import a BDF into C++test

1. Start console (or reuse this for BDF generation).
2. Set environment for the appropriate toolchain as described in the documentation (executables on PATH, EW_DIR envar).
3. Start C++test.
4. Follow steps of importing the BDF file described in documentation (see Importing project using Build Data File with the GUI wizard).
5. Confirm project settings correctness in **Properties> Parasoft> C++test> Build Settings**.

You must repeat the entire process following any changes to the original project unless you are adding new sources in existing source locations.