Integrating Raisonance 8051 Tools into the Cygnal IDE

1. Introduction

This application note describes how to integrate the Raisonance 8051 Tools into the Cygnal IDE (Integrated Development Environment). It applies to Version 1.61 or later of the Cygnal IDE. Integrating Raisonance 8051 Tools into the Cygnal IDE provides the most efficient development environment since compose, edit, and build operations are integrated in the same program as download and debug operations.

2. Key Points

- The Intel OMF-51 absolute object file generated by the Raisonance 8051 tools enables source-level debug from the Cygnal IDE.
- Once Raisonance Tools are integrated into the IDE they are called by simply pressing the 'Assemble/Compile Current File' button or the 'Build/Make Project' button.
- The generation of debug information and object extensions must be enabled in the Raisonance 8051 tools to support source-level debug using the Cygnal IDE.

3. Create a Project in the Cygnal IDE

A project is necessary in order to link object files created by your assembler/compiler and build an absolute 'OMF-51' output file. To create a project:

1. Open all assembly and/or 'C' source files which should be part of the project.
2. Under the 'Project' menu select 'Save Project As...'.
3. Enter a project workspace name, and click the 'Save' button.

4. Configure the Tool Chain Integration Dialog

Under the 'Project' menu select 'Tool Chain Integration' to bring up the dialog box shown below.
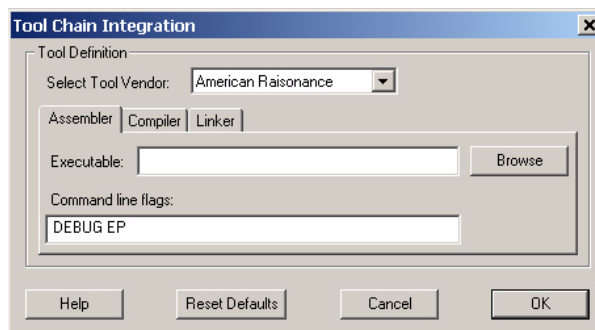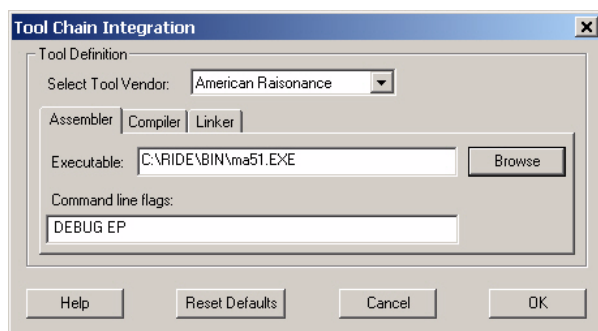

4.1.Tool Definition

You may define the Raisonance assembler, compiler, or both, but you must always define the linker to create an absolute object file.

1. Select Raisonance from the 'Select Tool Vendor' drop down list. Once Raisonance is selected, the default Command line flags for each tool are automatically filled in.
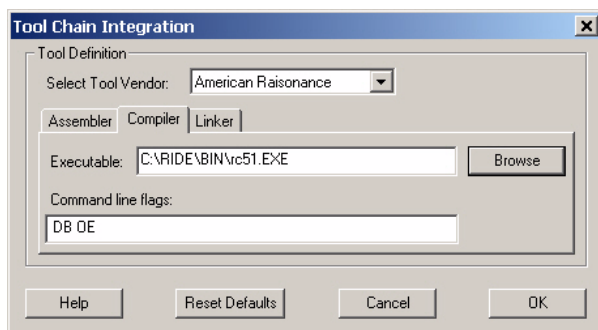

4.1.1 Assembler Tool Definition

1. Under the 'Assembler' tab, if the assembler executable is not already defined, click the browse button next to the 'Executable' edit box, and locate the assembler executable. The default for Raisonance MA51 is located at "\RIDE\BIN\MA51.EXE".
2. Enter any additional command line flags directly in the 'Command Line Flags' box.
3. The 'Assembler' tab with the default Raisonance settings should now look like this:

4.1.2 Compiler Tool Definition

1. Under the 'Compiler' tab, if the compiler executable is not already defined, click the browse button next to the 'Executable' edit box, and locate the compiler executable. The default for Raisonance RC51 is located at "\RIDE\BIN\RC51.EXE".
2. Enter any additional command line flags directly in the 'Command Line Flags' box.
3. The 'Compiler' tab with the default Raisonance settings should now look like this:

4.1.3 Linker Tool Definition

1. Under the 'Linker' tab, if the compiler executable is not already defined, click the browse button next to the 'Executable' edit box, and locate the linker executable. The default for Raisonance LX51 is located at "\RIDE\BIN\LX51.EXE".
2. Enter any additional command line flags directly in the 'Command line flags' box.
3. The 'Linker' tab with the default Raisonance settings should now look like this:
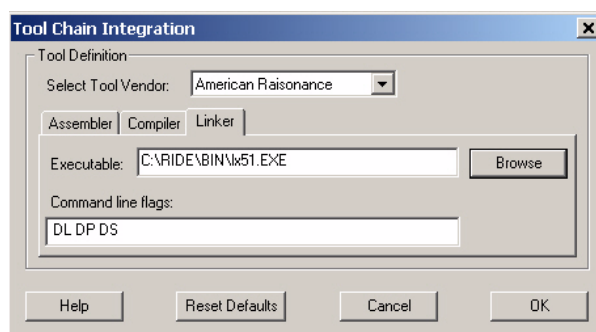
5. Target Build Configuration

Under the 'Project' menu select 'Target Build Configuration' to bring up the dialog box shown below.
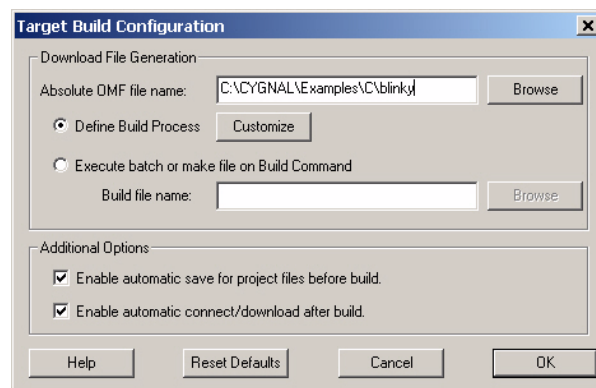
5.1. Download File Generation

First you must specify an output/download file. Define an output file, for the Raisonance linker to generate, in the following fashion:

1. Click the browse button next to the 'Download file name' edit box. Select a path and enter an output filename, "Blinky", for example.
2. Make sure the 'Define Build Process' button is selected.
3. Click the Customize button to bring up the 'Build Button Definition' dialog box. Default assemble, compile, and link selections will be made automatically, but make sure that all files have been correctly included in the build process.

5.1.1 Build Button Definition

The Build Button Definition Dialog allows you to select which files are to be included in the build process. Under each tab, you may add Files to assemble, compile, or link by selecting the desired file and clicking the 'Add' button. Files are removed in the same manner.
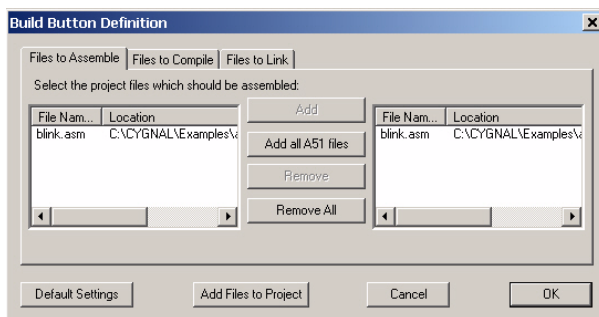
5.2. Additional Options

1. If the 'Enable automatic save for project files before build' box is checked, then all files included in the project will be saved automatically when the 'Build/Make project' button is pressed. The previous revision of your file(s) will be saved in a backup file. Backups are saved as the name of the file with the extensions #1, #2, #3, and so on up to the number of backups (N) created and available. '#1' being the most recent and 'N' being the least recent.
2. If the 'Enable automatic connect/download after build' box is checked, then the project will be downloaded to the target board automatically when the 'Build/Make project' button is pressed.

6. Building the Project

To compile and link all the files in the project, click the 'Build/Make project' button.

To assemble or compile just the current file, click the 'Assemble/Compile current file' button.

Errors and warnings generated during the build process can be viewed in the 'Build' tab of the Output window (typically found at the bottom of the screen). Double-clicking on an error that is associated with a line number will automatically move the cursor to the proper line number in the source file that generated the error.