Getting Started with your 8051 Project

TASKING
Embedded software development from Altium

The TASKING 8051 Software Development Toolset is based on the Embedded Development Environment (EDE) that incorporates a professional editor, project builder and debugger. From one toolbar you can compile a single file, build a project and start the CrossView Pro debugger.

Now that you've installed the demo, here is a brief overview of the toolset that will give you some assistance getting started with your project. If you have not already done so, we recommend that you read the *Getting Started with your 8051 Demo* tutorial to get acquainted with the first steps necessary to get the 8051 toolset running.

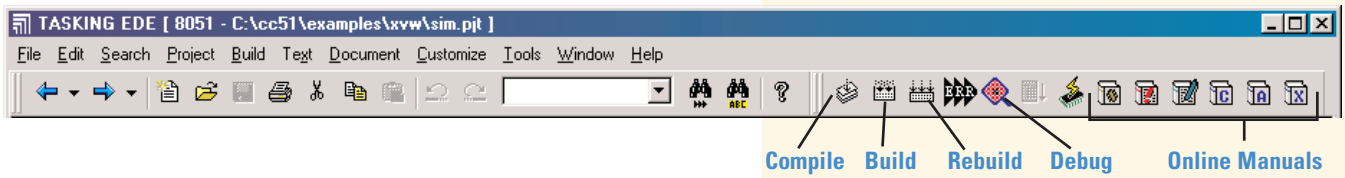
This document also contains an overview of some of the advanced features available within the EDE. These features demonstrate to you the power and versatility of the TASKING 8051 Software Development Toolset.

While walking through this Getting Started document you can also refer to the on-line manuals for further assistance. The on-line manuals are available as buttons in the toolbar or via the manual selection in the EDE menu.

DEFINING THE PROJECT SPACE

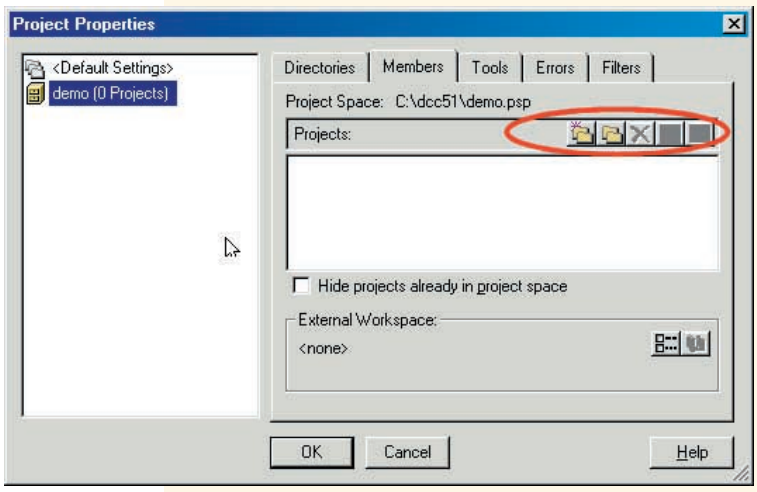
So let's get started with the 8051 toolset. If you are beginning with the development of a new application, you need to define the Project Space that will incorporate the one or multiple Projects that are part of it. There is a more extensive explanation on Project Spaces and Projects and how they relate, further on in this document; for now you can just follow the next instructions.

1. Select **File > New Project Space** from the menus. This will open the **Create a New Project Space** dialog box.



2. Now give your *Project Space* a name (for example, `demo`) and click **OK**. The **Project Properties** dialog box will appear for you to identify the *Projects* (new or existing) to be added. The properties will be saved in a file, for example `demo.psp`

3. There are two folder icons located adjacent to the **Projects** section within the **Project Properties** dialog box. Click on the **Add new project to project space** button which will then will open the **Add New Project to Project Space** dialog box.



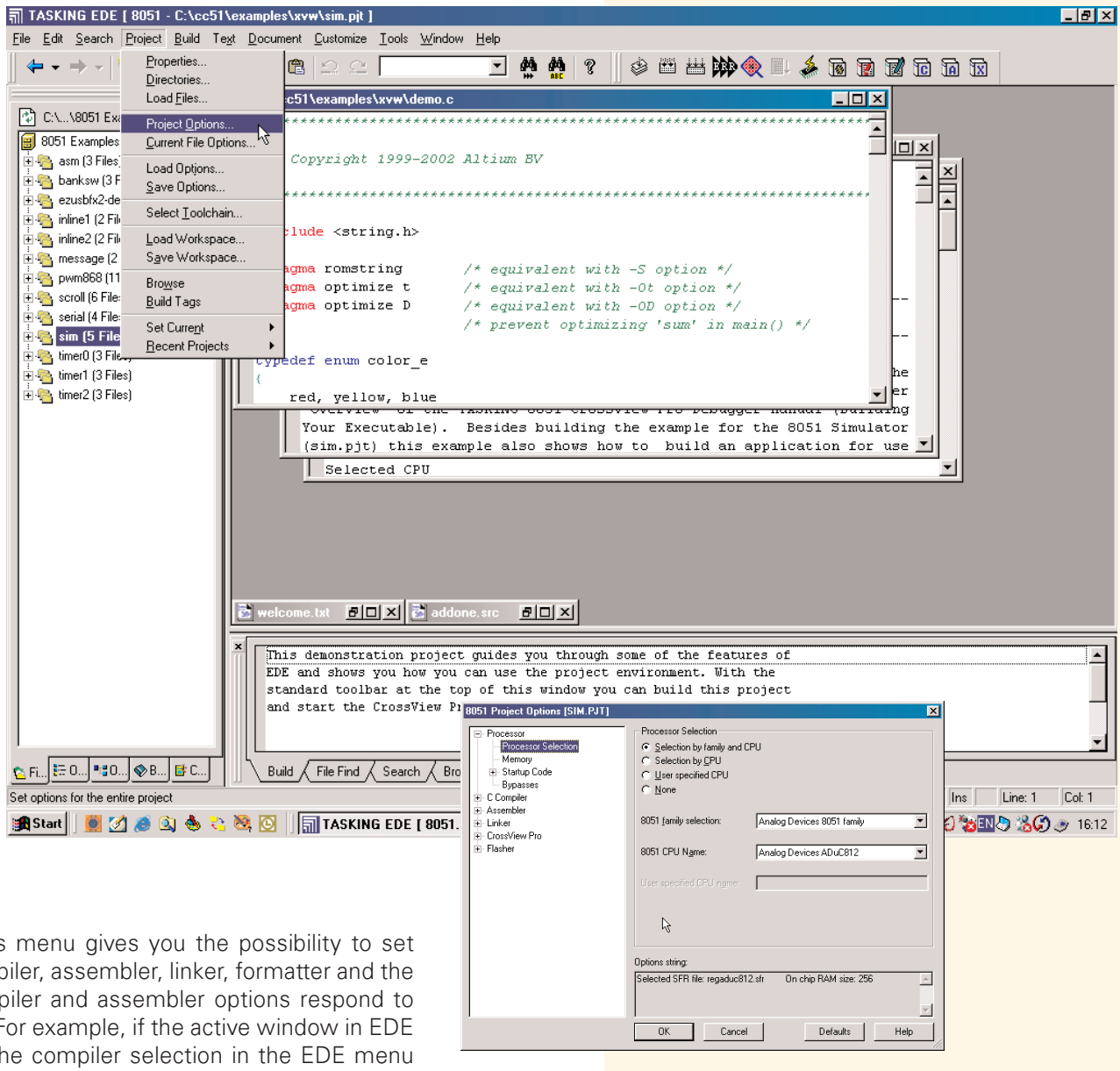
4. Give your project a name (for example, `control`) and then click **OK**. A file with the name `control.pjt` will be created. The **Project Properties** dialog box then appears for you to identify the files to be added.

5. Add all the files (source files, header files, text files describing your application, etc.) you want to be part of your project. If you do not have any source files yet, click on the **Add new file to project** button in the **Project Properties** dialog box. This will open the **Add New File to Project** dialog box. Now enter a new filename and click **OK**. A new empty file will be created and added to the project.

6. If you do have the source files already, click on the **Add Existing File to Project** button (third from the left) in the **Project Properties** dialog box. The **Select One or More Files to Add to Project** dialog box opens. Select the directory that contains the files you want to add to your project and add the applicable files by double-clicking on them, or by selecting them and pressing the **Open** button. Select **OK**. The new project is now open.

   Note: EDE automatically creates a *startup file* (discussed later in more detail) and a *makefile* for the project. EDE updates the *makefile* every time you modify your project.

7. After you've defined the files that belong to your project, the next operation is to select the target CPU you need to create your project for. To do this, select **Project > Project Options > Processor > Processor Selections** from the menu.



The Project Options menu gives you the possibility to set options for the compiler, assembler, linker, formatter and the debugger. The compiler and assembler options respond to the active window. For example, if the active window in EDE is a C source file, the compiler selection in the EDE menu allows you to set options for that specific file or for all C files in the project. The compiler's default options are set in the related panel (as it is for the assembler and the linker), so that you need to review, and eventually configure, the flags according to your requirements. For further assistance, consult the on-line manuals.

8. Once the characteristics of your project have been defined in terms of compiler settings, memory addresses, code locations, data locations and the format of the output file to generate, you can Build it by clicking on the **Rebuild** button available in the EDE toolbar.

9. After the files have been processed, the resulting messages generated by the tools will be displayed under the **Build** tab in the **Output** section at the bottom of the EDE window.

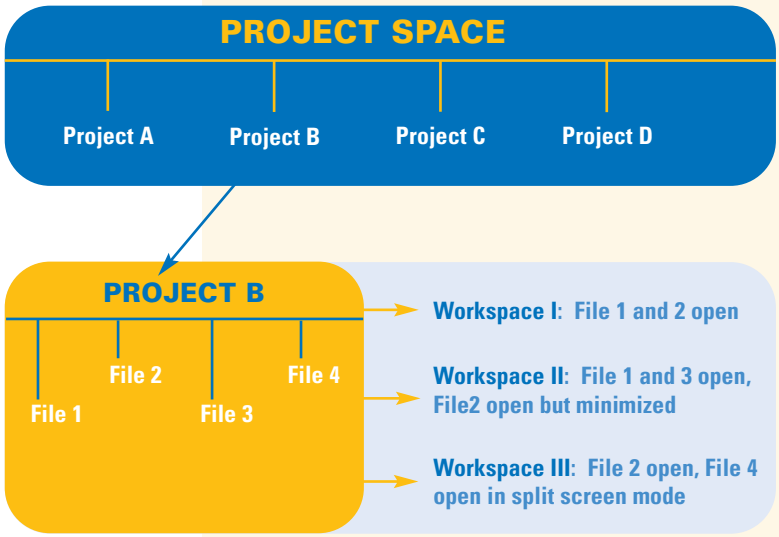
The generated *absolute* file (`.abs`) is now ready to be debugged with the **CrossView Pro Debugger**. See the *Debugging your Application with CrossView Pro* tutorial for more information.

USING ADVANCED FEATURES

Although all features of the EDE and the compiler tools are described extensively in the manuals, we would like to highlight some of the more unique and helpful features in this Getting Started document.

Project Spaces, Projects and Workspaces

Project spaces store sets of projects. Before a project can be made, a project space has to be set up. Project spaces allow multiple projects to be displayed at a time. If an existing project is opened without an existing project space, a project space is created automatically to house the project.



A project, at a minimum, is a list of files that you find it useful to group together logically. A project must be part of a project space.

Creating a project facilitates operating on the files in the project as a group, whether using version control, creating a "Tags" database, or just loading files. The files within the project may be further divided into *Workspaces*, which may contain project and non-project files.

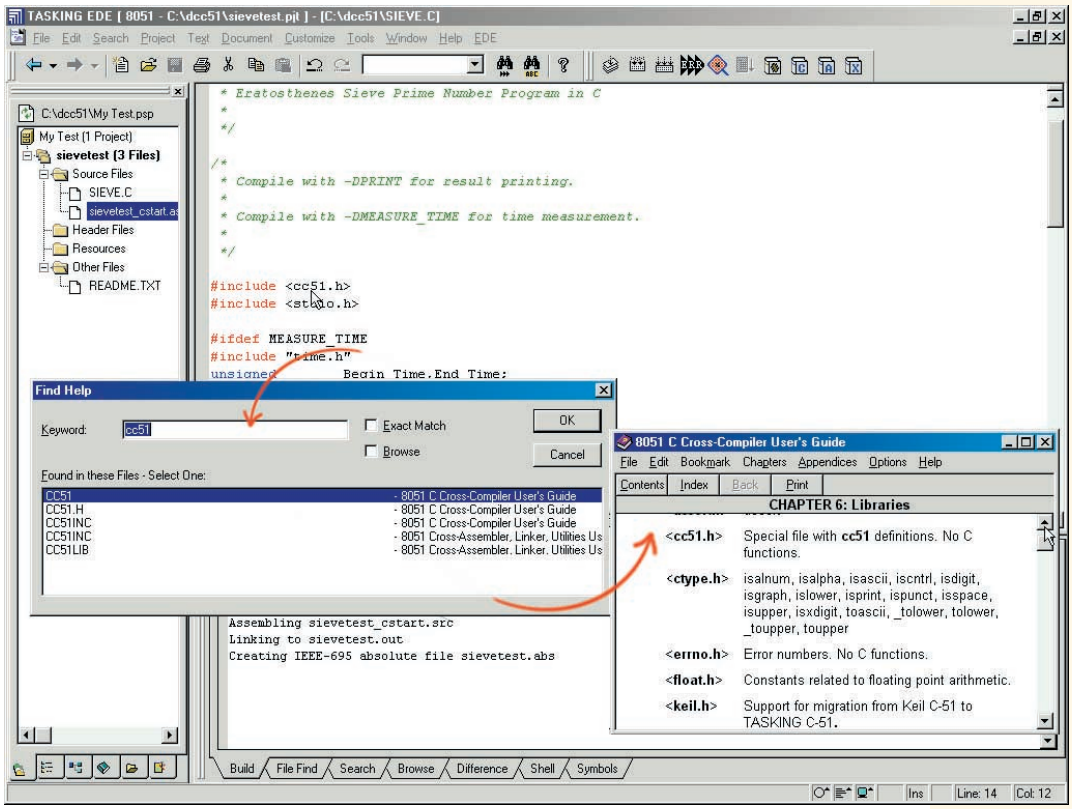
Within a project you can create a *Workspace*, in which you define the files that are open. So this helps you to focus on those parts of your project that you think are important.

You can think of each workspace almost as a separate instance of the **EDE**. That is, when you change workspaces you have the ability to pick up where you left off with a group of files. It doesn't matter if you worked on that group of files a half-hour before or three months ago. It is as if an instance of the **EDE** were frozen in time containing these files.

A workspace differs from a project in that it does not store the system-wide options normally stored in a configuration file. Only the transient options and settings are saved as part of the workspace. It retains information about the currently open windows and documents. Other state information is stored as part of the project.

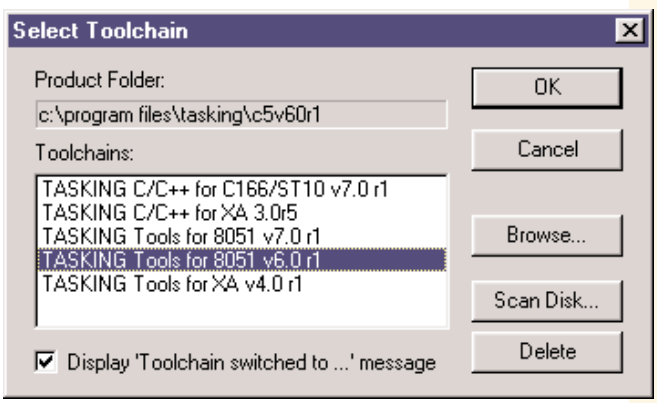
F1 Help!

There are many resources included in the package with additional information on the compiler, the EDE/editor, the debugger, etc. The documentation is available as HTML, PDF or Windows Help files . Each format has its own advantages, but the beauty of the Help files is that you can press the **F1** key anywhere in your code and when the help system recognizes the (key) word under your cursor, it will prompt you with a list of the manuals where it has found references to this keyword. A few clicks will then bring you quickly to the right spot in the documentation.



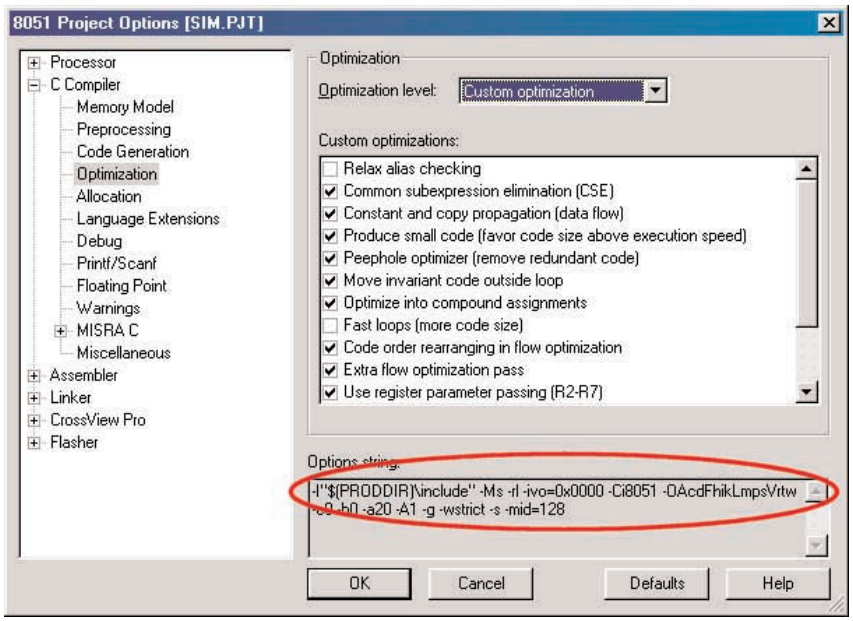
Switching Compilers

Are you involved in more than just one embedded development project and using more than one single software development toolset? From within EDE (**Project > Select Toolchain > Scan Disk**) you can scan your workstation for other TASKING toolsets and instantly switch over from one toolset to the other. This is not only very productive when you want to switch from 8051 to XA, but also if you want to switch from your current 8051 project that uses the latest TASKING C51compiler, to your previous project that needs a minor modification andfor which you used an older version of the compiler.



Tool Options

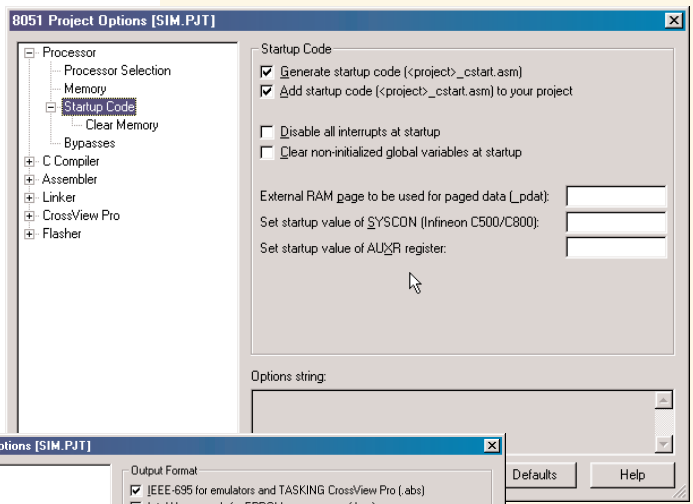
Being a command-line tool die-hard (hey, nothing's wrong with that!!), you may want to use the TASKING tools without the EDE/editor. Or you might have your own well-beloved editing environment that you feel comfortable with. No problem at all! The TASKING tools can be fully manipulated from the command line, and the documentation gives you a full overview of the



options to be used. As a matter of fact, from within EDE you can even see which options are passed on to the compiler and assembler/linker.

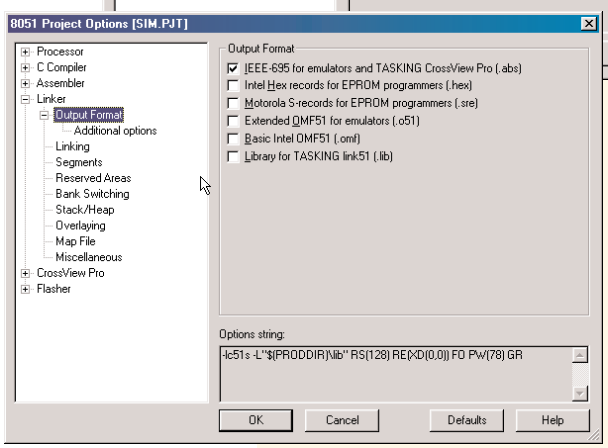
Automatic Generation of Startup Code

Of every embedded application the run time environment needs to be specified in the application's startup code. TASKING's EDE will automatically generate the startup code and save you from this tedious task. The file `<project>_cstart.asm` will be generated in the project directory and will also be added automatically to the project. Both generation as well as addition of the startup code file to the project can be switched on and off under **Project > Project Options > Processor > Startup Code**.



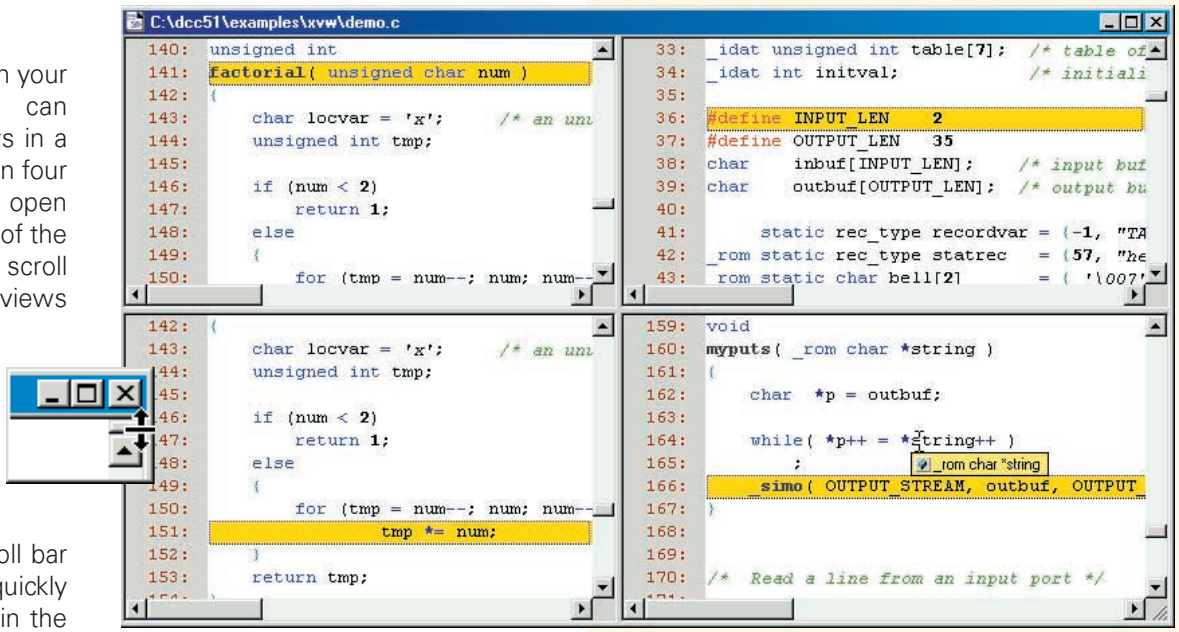
Output Formats

Upon completion of your embedded application you can have output files for different purposes be generated. In case you want to validate or debug your code, you can use the IEEE-695 or OMF51 formats. IEEE-695 is the format that the TASKING CrossView Pro debugger uses, but it is also the preferred format for all leading third party in-circuit emulators. For burning you code into an EPROM you may want to use the S-Rec or HEX formats.



Splitting Windows

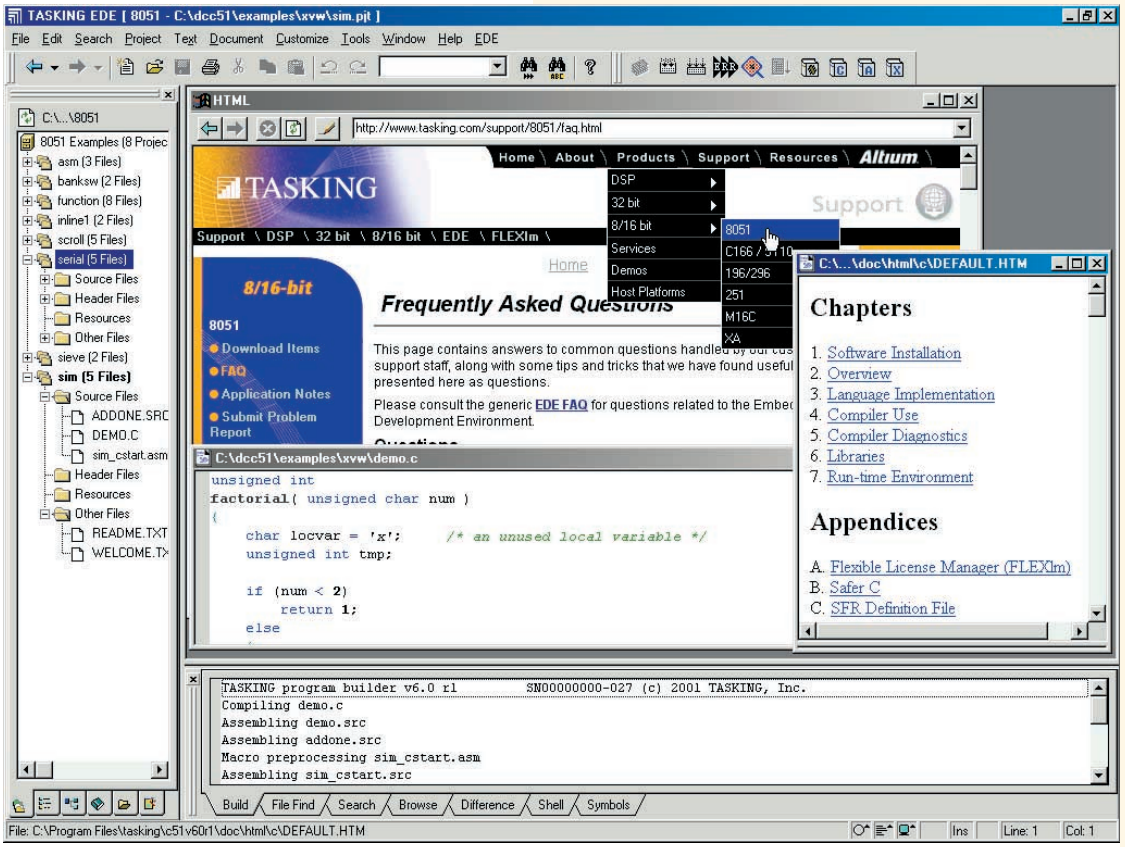
Instead of having one view on your source code file, you can manipulate the EDE windows in a way that you have two or even four different views on one single open file. You can enter data in any of the open views and by using the scroll bars you can reposition the views independently of each other. To split your source windows you simply pull one of the markers in the vertical or horizontal scroll bars.



By default the horizontal scroll bar is not activated, but you can quickly change that by right-clicking in the windows, then selecting **Properties > Window > Horizontal Scroll Bar**. When you have done this, you will see the split window marker to the left of the scroll bar.

Viewing HTML files, browsing the web

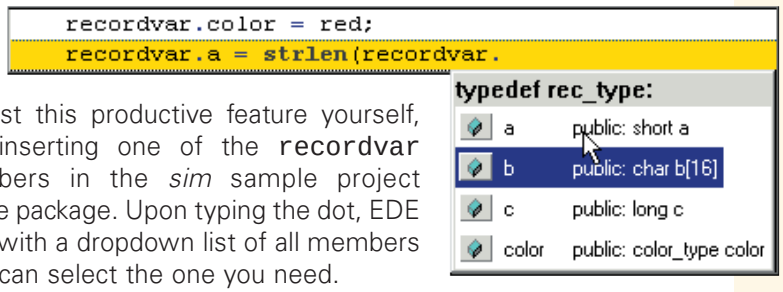
In EDE you can not only view an HTML formatted file, but you can even open a window and have it act as a browser. You can open HTML files on your local hard drive, or surf the web and browse through the TASKING Technical Support Pages for example. The screenshot above shows you the HTML version of the compiler manual in a small window and the larger window in the background presents the 8051 FAQ list on our web site. So everything at hand, in one seamless environment, to develop your embedded projects.



To enable the HTML viewer and web browser you need to make some new settings in EDE: From the menu, go to: **Customize > Libraries**. Here you can add extra functionality to EDE such as an FTP file transfer utility or an extended Clipboard viewer. Make sure that you select the **HTML WYSIWYG Editor/Viewer** in the *CodeWright Libraries* Section at the top, and add the file **CWIEXPLR.DLL** in the User Defined Libraries section in the middle. This DLL file can be found in the `\bin` directory of your tools installation. Once you have made these additional settings you can open an HTML file and an additional browse window will open automatically.

CodeSense - Automatic Word Completion

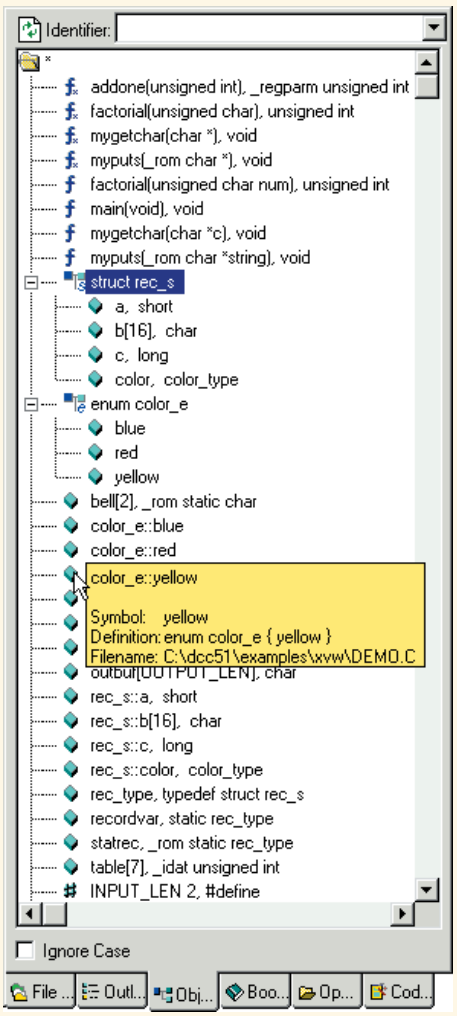
CodeSense uses tooltips and drop-down lists to complete the names of symbols (member lists, parameters, function definitions, and other source code elements). **Go-to** buttons are also available that allow you to quickly browse for symbols and symbol definitions. CodeSense information can be obtained from source code in the current open file, the current project, or from source code files in custom-specified CodeSense libraries. CodeSense hover tooltips and drop-down lists will display automatically at appropriate times while editing, or with the press of a keystroke. To test this productive feature yourself, you can start inserting one of the `recordvar` structure members in the *sim* sample project provided with the package. Upon typing the dot, EDE will prompt you with a dropdown list of all members from which you can select the one you need.



Source and Project Navigation

EDE offers you many ways to navigate through your project and source code. The project toolbar at the left side of your screen opens by default in the **Project Management** view, but through the tabs at the bottom you can open for example the **Bookmark Manager**, the **Outline Window**, the **CodeFolio Manager** or the **Object Browser**. The latter gives you not only an overview of all objects (variables, functions, defines, etc.) in your source file, but by selecting an object the focus of the source window instantly goes to the spot where the object is defined.

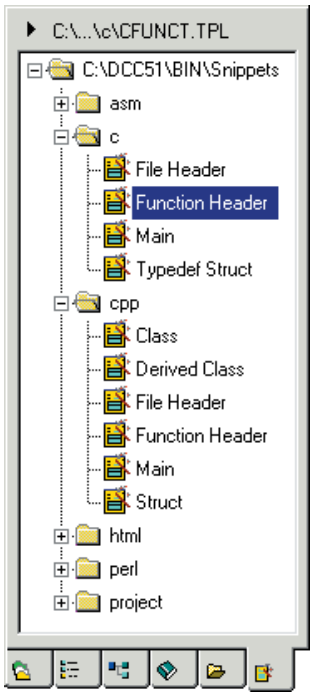
As an exercise you may type "`r*`" or "`*`" in the identifier entry field at the top, while you have the file `demo.c` of the *sim* project open. The result should be equal or similar to the screenshot on the right.



CodeFolio - a.k.a. Code Snippets

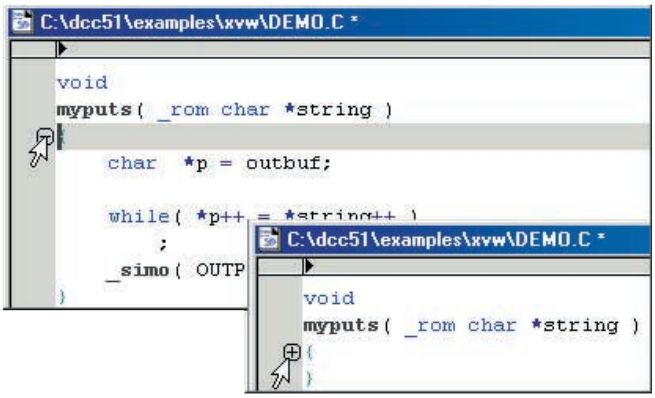
To help you using your company's standard templates for source code and source files, EDE offers you the **CodeFolio** feature. **CodeFolio** manages your templates and frequently used pieces of personal code (also known as *snippets*.) You can create a snippet by selecting a part of your code and drag it into the **CodeFolio** manager, where you can drop it in the directory of your choice. Once you have created a collection of snippets (we provided a few standard templates with the package) you can double-click a snippet to have it inserted at the cursor position in the open source file.

CodeFolio snippets may be extended with special **CodeFolio** macros; **CodeFolio** macros will automatically expand the current date and time, move the cursor to a specific location, perform mathematical equations, and more.



Selective Display, Collapse/Expand code

From the **Text > Selective Display** menu you have many options to gain a better overview of your source code. In the graphic the **Multi-level** option is used, which let your functions collapse at the braces level. Clicking on the

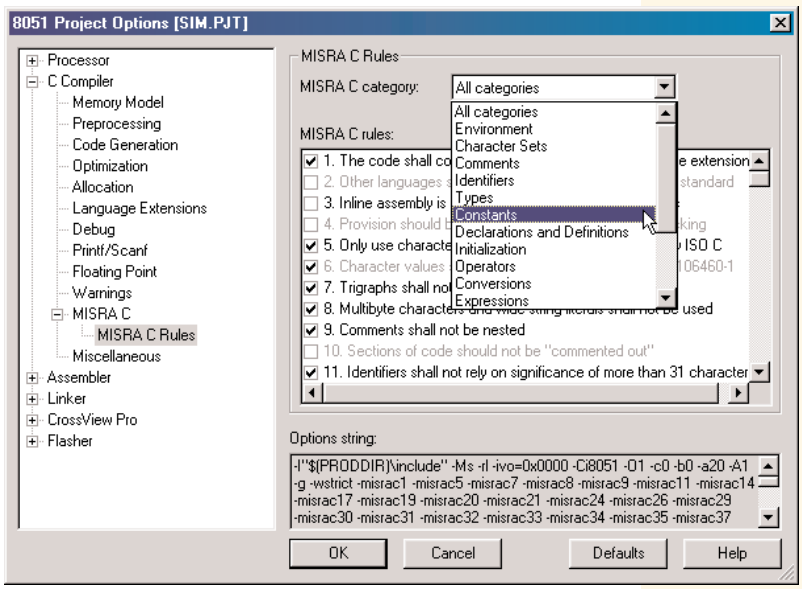


(+) or **(-)** in the left column will expand, respectively collapsing your code. This is just one of the six formatting options to help you keeping the right focus on your code.

MISRA C Support

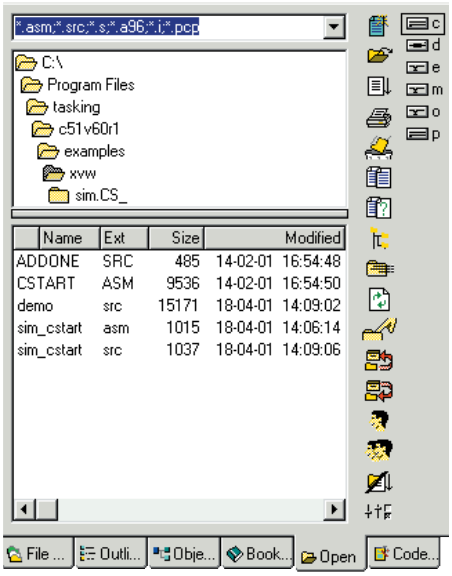
Are you concerned about the quality of your code or the reliability of the application that you develop? Does your product have to comply with safety related standards? The TASKING 8051 compiler is the only C51 compiler offering support for the MISRA C guidelines through the MISRA C options from the **Project > Project Options > C Compiler > MISRA C** menu. The Motor Industry Software Reliability Association (MISRA) is a consortium of companies that developed guidelines on which TASKING's MISRA C code checking is based. The consortium's effort was in response to the UK Safety Critical Systems Research Program, and the result is 127 programming rules applicable when developing safety-related applications in C.

Not every programmer is fully aware of the effects of all the possible constructs in the C language. For instance, the lack of type checking and the application of implicit casts tend to cause confusion and hence errors. A number of the features in C have not been well defined or are defined differently from what a programmer expects. You can use TASKING's MISRA C enhanced code checking to avoid falling in the pitfalls of the C language.



File Manager

The EDE does have a file manager integrated to navigate through the files of your project. Maybe that doesn't sound too exiting, as there are already many ways provided with Windows to browse your disks or network drives, but this manager offers you some extras that particularly are of benefit in a code development environment: You can for example Touch and Check in/out files, add your own User functions, or define the directories that you often visit. Give it a try! You'll like it.



And more…

A few other interesting features that you may want to try yourself:

- Right-click in a window or button bar and browse the list of applicable options
- The match brace facility to outline the scope of e.g. a function
- The file difference utility
- Integrate your company's source code version control system into EDE
- Setting local and global bookmarks
- Extend the functionality of EDE by activating libraries from the **Customize > Libraries** menu. A vast collection of additional libraries is available on the Internet, e.g. at *http://www.starbase.com/support/CodeWright/AddOns*

664-TCGEETG-5