

HIGHLIGHTS

- Total integrated development environment
- Easy project setup and management
- Can be tailored to your own environment
- Highly optimizing compiler
- Memory bank switching support
- MISRA C enhanced code checking
- Multiple data pointer support
- Basic and advanced debugging
- Kernel aware debugging
- Seamless integration with other TASKING toolsets
- Available on PC/Windows and Sun/Solaris

THE TASKING 8051 SOFTWARE DEVELOPMENT TOOLSET

The TASKING Software Development Toolset for the 8051 architecture provides a complete and cost-effective solution for programming the 8051 family of microcontrollers. The complete toolset consists of a C compiler, Intel compatible assembler, linker/locator, librarian, CrossView Pro debugger, and the TASKING EDE – our Embedded Development Environment that provides a composite interface to the complete toolset.

EMBEDDED DEVELOPMENT ENVIRONMENT

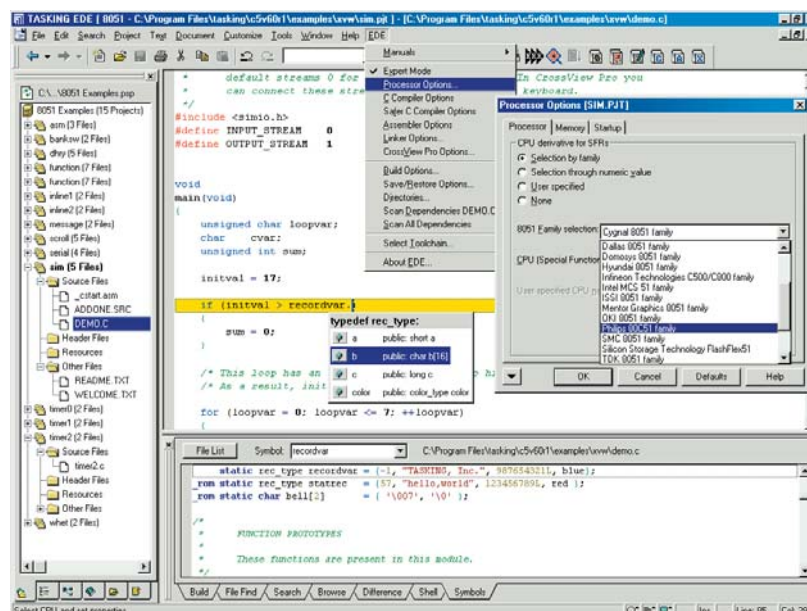
With the TASKING EDE you can create and maintain projects easily so your application is always up to date. All aspects of a project are saved in a project file, such as the source files that make up your application, the tool options (compiler, assembler, linker/locator, CrossView Pro debugger), the tool directories, and the options that describe the building process.

File dependencies as well as the sequence of operations required to build your application are handled automatically. The EDE offers you a range of very productive features for application and code development:

- **Easy/Expert modes** allow simplified configuration of the TASKING tools and the 8051 target processor for the less experienced user. After switching to Expert mode, all advanced options become available.
- **Project Spaces** allow grouping of multiple projects in one view, thus offering project management for more complex developments.
- **CodeSense** advanced coding assistance offers rich type-ahead features, which help you to select the next expected function parameter or available structure members. When positioning your mouse pointer over a function name, the function prototype will be displayed.
- **Tags Browsing** offers a graphical overview of the application's cross references and allows easy navigation through the available variables and functions.
- **CodeFolio** offers easy insertion of template code, thus adding to coding efficiency and consistency. It allows macro expansion and prompted input as you insert the code.
- **Right-Mouse-Button clicks** expedite a variety of tasks within the EDE (e.g., creating new files, adding files to a project, etc.).
- **HTML View Window** has been integrated to allow browsing through the product manuals, your project or code documentation or even surfing the internet.

8051 manufacturers supported

- | | |
|---------------------------------|-------------------------------------|
| ■ Acer Laboratories | ■ Intel |
| ■ Aeroflex UTMIC | ■ OKI Semiconductor |
| ■ Analog Devices | ■ Philips Semiconductors |
| ■ Atmel | ■ Silicon Storage Technology |
| ■ Atmel Wireless & μControllers | ■ Standard Microsystems Corporation |
| ■ Cygnal Integrated Products | ■ TDK Semiconductor Corp. |
| ■ Cypress | ■ Texas Instruments |
| ■ Cypress/Anchor Chips | ■ Triscend |
| ■ Dallas Semiconductor | ■ Winbond Electronics Corp. |
| ■ Damosys | ■ Xicor |
| ■ Hyundai Electronics | |
| ■ Infineon Technologies | |
| ■ Integrated Silicon Solution | |
- A complete overview is available on our website.*



- **XML collapsible grid viewer** displays a hierarchy of the elements and element attributes in an XML document.
- **Split windows** provide full control over source code by allowing you to split your file horizontally or vertically into as many as four interactive edit windows.

See the following box for more highlights of TASKING's award-winning EDE.

POWERFUL EDITING

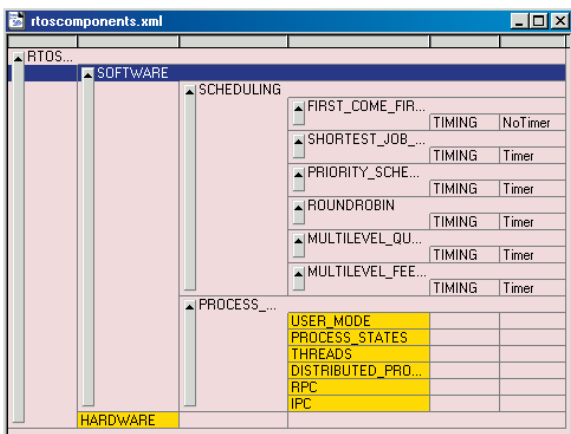
- Hexadecimal editing
- Compile/track errors
- Background symbol compilation
- Print preview
- Difference editing for side-by-side code comparisons
- Syntax highlighting
- Smart indenting
- Multiple clipboards and scrap buffers. "Clip View" to display the contents
- Re-definable keyboard
- Keystroke recording and playback
- Run a command prompt in a buffer
- Configuration Wizards to walk you through several editor features
- Select text by stream, column, or line
- Drag-and-drop to load files from Explorer
- File size and scrap buffer up to 2GB
- Maximum number of buffers limited only by memory and disk size
- Spell check recognizes comments and string constants
- True soft word wrapping (without reformatting your code)
- Customize menu, toolbars, and pop up menus
- Automate your processes using macros
- Version control interface to various standard products
- CUA, BRIEF, Epsilon, and Vi key maps

C COMPILER

Unlike other 8051 C compilers, the TASKING C51 compiler has been designed and built specifically for the 8051 microcontroller architecture. This means that you can access all the special features of the 8051 in C without violating any ANSI standards. Special features include multiple address spaces (with full pointer support), bit memory, special function registers (I/O ports), interrupt functions using bank switching, user in-line C functions, (create your own in-line functions) and much more.

General features of our C compiler include:

- **Full ANSI C to ensure early error detection**
- **Complete 8051 family support, desired derivative is switch selectable**
- **In-line functions and in-line assembly**
- **Single precision floating point**
- **Generates Intel-compatible assembly source**
- **Outputs symbol information for debugging**
- **Complete C and run-time library support**
- **Generates reentrant and relocatable code and data**
- **Intel OMF-51 and IEEE-695 object format to ensure interoperability with debuggers and emulators**
- **Intelligent configuration of startup code**
- **User-controlled mapping of code and data**



Altium's TASKING compilers are tested for ISO or ANSI conformance against authoritative validation suites such as Plum Hall and Perennial. Additionally, the optimization techniques of the compilers are tested with various large real-world applications as well as industry benchmark standards such as Nullstone and EEMBC.

MISRA C

Based on the "Guidelines for the use of the C language in vehicle based software" published by the Motor Industry Software Reliability Association (MISRA®), Altium is the first to implement the MISRA C concept in a software development environment. MISRA C guides programmers in writing more robust C code by defining selectable C usage restriction rules. Through a system of strict code checking, the use of error-prone C-constructs can be prevented.

A predefined configuration for compliance with the MISRA C guidelines is available with a single click. It is also possible using pull-down menus to enable a custom set of MISRA C rules to suit specific company requirements.

This means that, under the guidance of MISRA C in the TASKING tool chain, you can write better, more maintainable code that utilises the benefits of high-level C programming without the inherent dangers.

MAKE YOUR TEXT SMART – source code is no longer text only

Insert button links in text files to perform special actions:

- **View related documents or diagrams**
- **Run macros and applications**
- **View categories and lists of bookmarks and links**
- **Create pop-up notes**

Your source code remains ASCII and compatible with other editors.

PRODUCTIVE EDITOR EXTENSIONS

Tailor the editor to your needs and wishes:

- **Integrated FTP utility**
- **Run macros and applications**
- **View categories and lists of bookmarks and links**
- **Create pop-up notes**

A wide range of various extensions available from fellow developers.

Powerful optimizations

The TASKING 8051 C compiler tools implement a wide variety of optimizations to allow reduction of code and data size as well as execution time. Optimizations can be applied on the complete project or specific files, or they can be switched on/off at function or source line level.

Optimizations include:

- **Various loop and jump optimizations to speed up execution and/or reduce code size.**
- **Common sub expression elimination detects and eliminates repeating (sub-) expressions.**
- **Common tail merging for finding duplicate sequences of code and merging them together to reduce code size.**
- **Dead assignment, dead storage and dead code elimination removes all kinds of unreachable code or invariant data.**
- **Peephole optimizations replace instruction sequences with equivalent but faster and/or shorter sequences, or delete obsolete instructions.**

Data types

All ANSI types are supported. In addition to these types, `_bit`, `_sfrbyte`, and `_sfrbit` are added. The keywords `_sfrbyte` and `_sfrbit` are available to access special function registers which deal with I/O. SFR's are treated like memory mapped variables declared with the volatile type qualifier.

Memory models

The C compiler supports four different memory models. The memory model determines the default memory type for variables declared with no explicit memory type, pointers and function parameters, and automatics. These models can be used separately or mixed (per function) to ensure the best possible code generation.

Parameter passing

The C compiler passes up to three parameters in CPU registers. You can control parameter passing with a command line option and for each function separately using the `_regparm` and `_cdecl` function qualifiers. If no registers are available for parameter pass, if too many parameters are involved, or if register passing has been disabled, the compiler automatically uses fixed memory locations or the virtual stack. The latter can be controlled with the `_small`, `_auxpage`, `_large`, and `_reentrant` function qualifiers which, if used, allow mixed memory model programming. Passing parameters in CPU registers in combination with mixed memory model programming significantly improves the performance of the application.

Built-in functions

Built-in functions generate in-line (faster) code to perform the library function and utilize certain 8051 instructions that do not have an equivalent in C.

Interrupt functions

C functions can be declared to serve as interrupt service routines. You can specify the interrupt number and the register bank selection via the `_interrupt` and `_using` function qualifier. The compiler emits the corresponding interrupt vector and the appropriate entry and exit code (using the RETI instruction).

Switch statement

The C compiler supports three different methods for implementing a switch statement. You can control how the compiler should optimize the switch statement with a `#pragma`. By default the compiler will choose the smartest implementation, depending on the amount and contents of the case statements.

Multiple datapointer support

The toolset has support for 8051 derivatives with multiple datapointers, such as the implementations from AMD, Dallas, Infineon Technologies, and Philips. To build an application for a specific derivative, the user simply selects this Multiple Datapointer derivative from a predefined list within the EDE.

Memory banking support

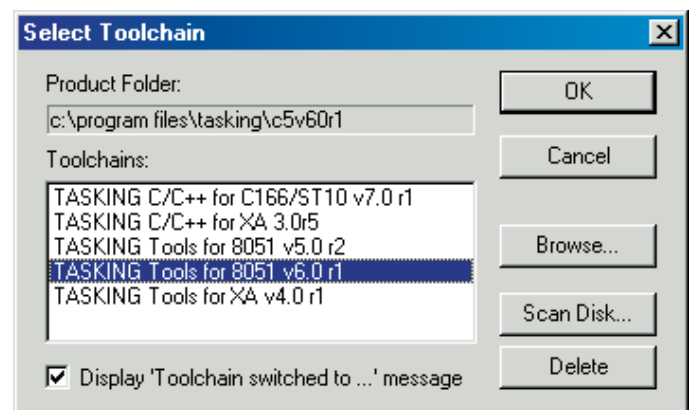
The compiler supports code memory banking, allowing up to 256 banks of 64Kb. Segments are automatically located over all available banks; the linker detects calls from one bank to another and inserts stub routines to handle the bank switch.

Libraries

The compiler package includes ANSI C libraries, run-time libraries including I/O calls (+ printf), memory management, arithmetic functions, and floating points for both internal and external RAM. The source code provided for most of the library routines allows you to tailor the libraries to your specific application.

Toolset Integration

The 8051 tools integrate seamlessly with other TASKING toolsets into one environment. Switching from one toolset (or version of a toolset) to another toolset is hassle free and can be performed instantly. This feature is very helpful if you want to use a specific version of a compiler in one of your older projects, or want to use a newer version in your next project. In cases where you want to upgrade your project to a different architecture, this integration eases the transition.



THE ASSEMBLER

The TASKING assembler is an integral part of the toolset and translates 8051 assembly language into relocatable object code. The assembler accepts Intel compatible assembler source programs and produces relocatable (.obj) object files. An absolute or executable load image is then obtained by using the linker/locator.

Features of the assembler include:

- **Production of relocatable object code and listing files**
- **Acceptance of Intel compatible source programs**
- **Compatibility with high-level and assembly-level debuggers**
- **Optimization of jmp/call instructions**
- **Segment overlay support at the assembly level**
- **Intel compatible macro preprocessor**
- **Extensive segment directives**
- **Error file with textual error reporting**

Linker/Locator

The linker/locator is an essential part of the software building process that enables you to configure the code to match your target environment. The linker/locator brings together all the necessary relocatable objects (including library modules), resolves external references, and then locates the modules in memory according to your specification.

Features include:

- Intel-compatible linker controls
- Object files and object libraries accepted in the Intel OMF-51 file format
- Generation of absolute Intel OMF-51 or IEEE-695 object files
- Automatic segment overlaying using call graph information from the compiler and assembler
- Absolute map files and diagnostic messages
- Automatic inclusion of library modules
- Map listing to help with debugging

CROSSVIEW PRO DEBUGGER

An easy-to-use interface with powerful and extensive debugging features helps you debug your applications faster. The CrossView Pro debugger is a true Windows application complete with multiple, resizable, and independently controlled windows. It combines the flexibility of the C language with the control of code execution found in assembly language, bringing functionality that reduces the amount of time spent testing and debugging.

Functionality includes:

- Bubble-Spy™ technology for quick and easy inspection of variables and functions
- Large Smartbuttons to maximize the viewable debugger workbench
- Tracking scope and monitoring of locals
- "Intelligent" source window
- Double click and right mouse button functions

You choose the windows you need to view the different aspects of your code during debugging.

Source window

The working window is the source window. It lets you view source; set and clear breakpoints, assertions and code coverage markers; monitor and inspect variables; search for strings, functions, lines and addresses; call functions and evaluate expressions; and view performance analysis data. The source window allows you to view your code at C level or assembly level, or you can choose a mixed mode that allows you to simultaneously view your C code intermixed with its corresponding assembly code.

From the source window, you can jump directly into the editor within the EDE, and you will find yourself positioned at the source line where you had your cursor in the debugger.

Multiple data windows

Data windows enable you to watch or show data, browse for locals or globals, double-click to modify values, or expand and contract complex data structures. Within these windows you can reformat (change display radix and type) on an element-by-element basis. You can show or watch locals from any stack level, automatically track and display locals, and easily copy any variable to a new window as show or watch.

Register window

The register window displays and modifies CPU register values. The window is fully configurable and is updated every time the program is stopped. Highlighted registers indicate what has changed since the last stop.

Stack window

The stack window displays the state of the current stack frame. With simple point-and-click operations, you can set up level breakpoints, display source for function calls, and display local variables for selected functions.

Multiple memory windows

The memory window with ASCII display enables you to monitor any address change, double-click to modify, and have complete control over the size and format of data.

Coverage and profiling

With coverage you can check whether the code of your application is reached (executed at least once) or not. Coverage helps you build a complete test suite for your product, which improves the quality of your application. The CrossView Pro debugger also supports coverage of data regions. Profiling allows you to analyze the performance of your application. You can see how the total time is divided among the C functions and which functions should be optimized for speed.

Software assertions

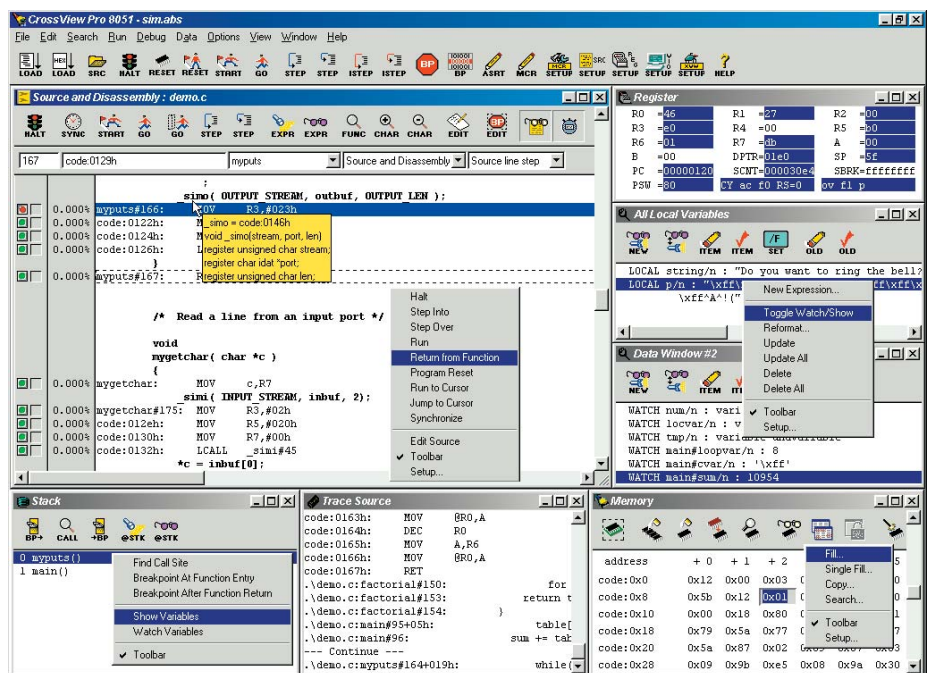
Software assertions let you execute user-specified command lists after running every line of source code. This is the software equivalent of data breakpoints, and can be used to set up sophisticated error checking mechanisms that uncover the most elusive of bugs.

C-Like macro language

With C-like macro language, you can read and/or modify application variables and call application functions from macros. It has full C expression syntax.

Programmable Data Analysis

Programmable Data Analysis enables quick detection of gross errors in your signal processing routines by reducing large sets of data into meaningful visual diagrams. The CrossView Pro debugger can analyze the data according to pre-defined or user-defined specifications, and display the data the way you need it. This eliminates the need for reviewing or post-processing large files of raw data. You can also view the same set of data in several ways at the same time (e.g., in the time and the frequency domains).

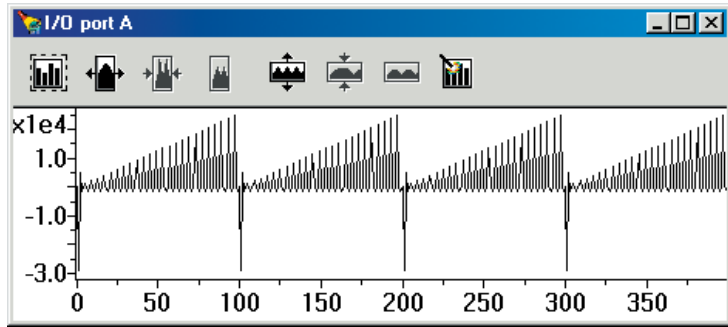


Multiple execution environments

The CrossView Pro debugger supports two execution environments, a ROM Monitor and a Simulator, with a standard user interface.

ROM monitor environment

The CrossView Pro ROM debugger can be used with any commercial off-the-shelf evaluation board or custom-developed target application.



The CrossView Pro debugger, running on a host computer system, communicates with the monitor on the target board via an RS232 interface using a very efficient protocol.

The resources used by the monitor program are kept to a minimum.

The monitor uses:

- 3Kbytes
- One register bank
- 20 bytes of stack space
- 12 bits of bit addressable memory

Simulator

The simulator environment allows you to test, debug, and monitor the performance of code in a known and repeatable environment independent of target hardware. It uses the same description file as the linker/locator when locating your application, so it therefore knows exactly where and how memory is mapped.

All CrossView Pro debugger features, including C level trace, Code/Data Coverage, performance analysis (profiling), and unlimited amount of code and data breakpoints, are available to you so you can test code before target hardware is available.

Open architecture

The CrossView Pro debugger supports a truly open architecture by providing public interfaces and supporting industry standards. Public interfaces such as the Kernel Debug Interface (KDI) and Generic Debug Instrument Interface (GDI) provide third party vendors easy access and interface to the CrossView Pro debugger framework. The KDI can also be used to provide kernel aware debugging for your "in-house" kernel! Visit our website for more information.

COOPERATION WITH THIRD PARTIES

Working with other suppliers of products for the 8051 architecture gives us the opportunity to improve the tools that we deliver and to create a total solution concept for your 8051 application development. A complete overview of third parties is available on request.

Emulators

TASKING's 8051 tools are supported by broad range of emulator manufacturers including, but not limited to, Ashling, Ceibo, Hitex, Lauterbach, Metalink, and Nohau.

Real-Time Operating Systems

Amongst the kernel manufacturers that offer a TASKING compatible kernel are CMX Systems (CMX) and Lineo (RTXC). With MicroNet™ CMX offers a TCP/IP solution for the 8051.

Evaluation boards

TASKING offers plug-and-play support for various commercial off-the-shelf evaluation boards. This allows the user to connect the CrossView Pro ROM monitor debugger via a serial connection to the evaluation board and monitor the application with CrossView Pro.

CUSTOMER SUPPORT

When you purchase a TASKING product, it is the beginning of a long term relationship. TASKING is dedicated to providing quality products and support worldwide. This support includes program quality control, product update service, and support personnel to answer questions by telephone, fax or email.

PRODUCT PACKAGING & ORDERING CODES

Each TASKING product comes with full documentation in binders. The documentation is available on-line as well and provides full-text search capabilities for quick and easy lookup of topics.

The 8051 Development Toolset is available for PC/Windows and Sun/Solaris.

Product Code	Package contents
07-200-008-024	EDE, C and MISRA C Compiler, Assembler/Linker, CrossView Pro ROM Monitor and Simulator Debugger

A trial version of the 8051 Software Development Toolset is downloadable from our website at:

www.tasking.com/8051

ALTIUM SALES OFFICES

North America - Altium Inc

3207 Grey Hawk Court, Suite 100
Carlsbad, CA 92010
Ph: +1 760-231-0760
Fax: +1 760-231-0761
Email: sales.na@altium.com

China – Altium Information Technology (Shanghai) Co., Ltd.

9C, East Hope Plaza
No. 1777 Century Avenue
Shanghai 200122
Ph: +86 21 6182 3900
Fax: +86 21 6876 4015
Email: sales.cn@altium.com

Australia – Altium Limited

3 Minna Close, Belrose
NSW 2085
Ph: +61 2 8622 8100
Fax: +61 2 8622 8140
Email: sales.au@altium.com

Germany – Altium Europe GmbH

Albert-Nestler-Straße 7
76131 Karlsruhe
Ph: +49 (0) 721 8244 300
Fax: +49 (0) 721 8244 320
Email: sales.de@altium.com

France – Protel AG

(A subsidiary of Altium Limited)
121 rue d'Aguesseau
92100 Boulogne-Billancourt
Ph: 0800 88 05 06
Fax: 0800 82 85 92
Email: info.fr@altium.com