

HIGHLIGHTS

- Fully-integrated embedded development environment
- Incorporates next-generation Viper compiler technology
 - Excellent code performance
 - ISO C'99 and ARM ABI compliant
 - MISRA support
 - Code profiling
 - Run-time error checking
- CrossView Pro debugger
 - Instruction set simulator
 - On-Chip debug
 - (OSEK) kernel aware
- Available for PC/Windows and SUN/Solaris

INTRODUCING THE TASKING C/C++ COMPILER AND DEBUGGER VX-TOOLSET FOR ARM®

The TASKING VX-toolset for ARM® brings to developers the power of Altium's sophisticated, next-generation Viper C compiler technology framework, allowing them to take full advantage of the highly-popular ARM architecture. With its Viper technology, the TASKING VX-toolset for ARM is able to generate code with the level of execution speed and code density needed for tomorrow's automotive, industrial and telematics applications. The compiler features up-to-date functionalities such as MISRA C code checking, profiling through code instrumentation and run-time error checking capabilities.

The toolset consists of:

- ISO C++ compiler, scalable to EC++
- C compiler, ISO C'99 compliant, with integrated 'MISRA C' enhanced code checking
- Assembler with macro-preprocessor
- C/C++ libraries, run-time libraries, floating-point libraries
- Linker and locator
- CrossView Pro debugger with two execution environments
 - Simulator
 - OCDS debugging over JTAG*

* Product planned, call for availability.

The C compiler and debugger are also included in Altium Designer, which is available separately from Altium. Altium Designer is the industry's first single, unified application that incorporates all the technologies and capabilities necessary for complete electronic product development. Altium Designer integrates board- and FPGA-level system design, embedded software development for FPGA-based and discrete processors, and PCB layout, editing and manufacturing within a single design environment.

ARM® ARCHITECTURAL SUPPORT

The toolset provides support for a wide range of ARM processors, such as ARM7™, ARM9™, ARM9E™ and ARM Cortex™-M based cores and standard microcontrollers, including devices with a Vector Floating Point (VFP) coprocessor. We continuously add ARM variants to the supported list, including specific derivatives from silicon vendors such as Analog Devices, Atmel, NXP, Samsung, Sharp and STMicroelectronics.

The compiler can generate code for Thumb® Mode and includes run-time libraries built specifically for Thumb Mode. ARM and Thumb code may be mixed in the same source file, allowing you to code your fast signal-processing algorithms and interrupt handlers in ARM Mode, while benefiting from the smallest code size in Thumb Mode.

The toolset's linker can generate small pieces of code (so called 'veneers') for ARM and Thumb interworking. These veneers enable smaller and more efficient programs:

- Switches between Thumb Mode and ARM Mode are inserted when required
- Long calls are automatically generated when required

The compiler generates code with either Big-Endian or Little-Endian byte order, putting the most significant byte at the lowest or highest address.

The TASKING CrossView Pro debugger is included in the ARM VX-toolset for debugging your program code. Through the supported ELF/DWARF object/debug standard you can also use other industry popular debuggers and emulators.

EMBEDDED DEVELOPMENT ENVIRONMENT

With the TASKING EDE for ARM, you can create and maintain projects with minimum effort. All project-related aspects, such as the application source files, the tool options (compiler, assembler, linker/locator, CrossView Pro debugger), file management and the options of the build process, are managed from one central point. File dependencies, as well as the sequence of operations required to build the application, are handled automatically.

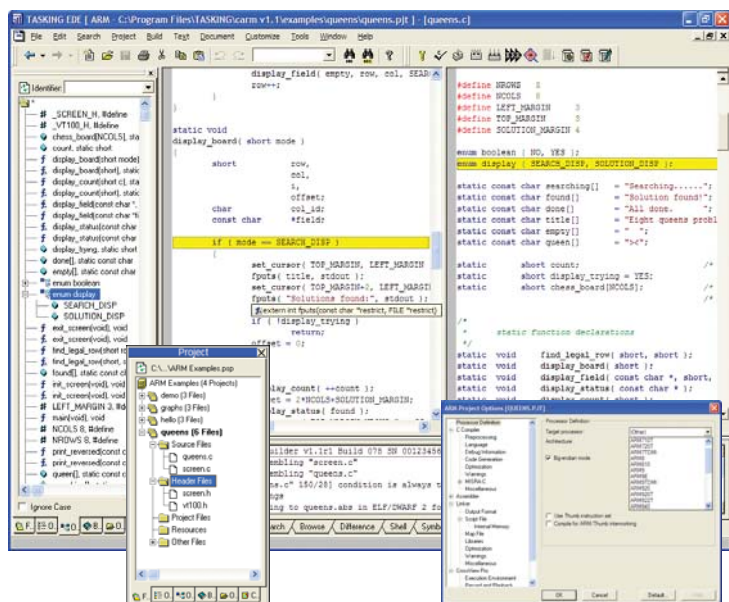
ARM processors supported:

- ARM7EJ-S™
- ARM710T™
- ARM720T™
- ARM7TDMI®
- ARM7TDMI-S™
- ARM9TDMI®
- ARM920T™
- ARM922T™
- ARM940T™
- ARM926EJ-S™
- ARM946E-S™
- ARM966E-S™
- ARM968E-S™
- ARM10TDMI™
- ARM1020E™
- ARM1022E™
- ARM1026EJ-S™
- ARM Cortex™-M1
- ARM Cortex™-M3
- StrongARM®
- StrongARM110
- StrongARM1100
- XScale®



The EDE of the ARM VX-toolset offers many productive features for application and code development, such as:

- **Project Spaces** that enable you to group multiple projects in one view, offering improved project management for more complex developments
- **CodeSense** advanced coding assistance that offers rich type-ahead features, assisting you in selecting the next expected function parameter or available structure member. When positioning your mouse pointer over a function name, the function prototype will be displayed
- **Tags Browsing** which offers you a graphical overview of the application's cross-references and allows easy navigation through the available variables and functions
- **CodeFolio** that allows you to easily insert 'snippets' of template code, thus adding to coding efficiency, making possible macro expansion and prompted input as you insert the code
- **ChromaCoding** provides syntax coloring for programming languages such as C, C++ and assembly
- **Split Windows** that provide full control over source code by allowing you to split your file horizontally or vertically into as many as four edit windows
- **Selective Display** gives you a better overview of your source code, allowing you to collapse and expand functions at the braces level for example. Six formatting options are available



C COMPILER

Based upon Altium's latest DSP-C compiler technologies, the VX-toolset C compiler is reliable, compliant, competitive, complete, easy to use and generates the most optimal code possible to allow you to take full advantage of the ARM architecture.

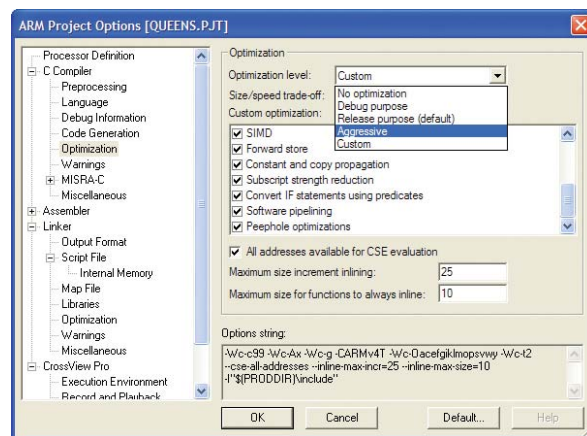
The TASKING VX-compiler for ARM is tested for ISO C'99 and ISO C++ conformity against authoritative validation suites, such as Perennial and Plum Hall. In addition, the optimization techniques of the compilers are tested with various large real-world applications (e.g. audio/GSM codec suites), as well as industry benchmark standards such as Nullstone.

Fast and compact

Altium understands that you expect your ARM compiler to produce the most optimal code possible with no fuss. With its Viper compiler technology, the TASKING VX-toolset for ARM, in its default configuration, generates code with the smallest footprint and fastest execution possible. Depending on the specific requirements of your ARM application, optimizations can then be further tweaked for smaller code size or higher execution speed.

Compiler optimizations include:

- **Partial Redundancy Elimination (PRE)** detects and eliminates repeating (sub-) expressions
- **Various Loop and Jump optimizations** speed up execution and reduce code size
- **Control-flow and code-reduction optimizations** remove dead code and perform transformations to minimize jumps
- **Function inlining** replaces calls to small functions with inlined copies of the function code
- **Peephole optimizations** replace instruction sequences with equivalent but faster and/or shorter sequences, or remove obsolete instructions
- **Inter-procedural register allocation**



Code profiling

In addition to the profiling features built into the debugger, the compiler is equipped with a profiler that uses code instrumentation. Code profiling can be used to determine which pieces of your code execute slower than expected and which functions contribute most to the overall execution time of a program. A profile can also tell you which functions are called more or less often than expected. The advantage of this code profiling option in the compiler is that it can give a complete call graph of the application annotated with the time spent in each function and basic block.

Several forms of profiling output can be obtained:

- **Flat profile** – shows how much time is spent in each function, how many times that function has been called, and optionally how often each lexical block within the function is executed. This is very useful if you want to know which functions or lexical blocks consume most cycles
- **Call graph profile** – shows, for each function, which functions called it, which other functions it called, and how many times. There is also an estimate of how much time was spent in the subroutines of each function

Syntax and semantic checks

The compiler offers a vast array of syntax and semantic checks that warn about potential undesirable effects or bugs in your program. Early fixing of source code problems when reported by the compiler generally only takes minutes compared to hours, or days, when the problem is discovered at run time.

Examples of compile-time checks include:

- **Validating printf and scanf format strings** against the type of the actual arguments
- **Using uninitialized memory locations**
- **Detecting unused variables**
- **Value tracking, which is used to detect errors such as**
 - array subscript out of bounds
 - division by zero
 - constant conditions

Run-time error checking

TASKING's run-time error checking capabilities in the compiler offer a wealth of checks that reveal run-time errors when they first occur. The kind of errors found by run-time error checking are typically hard to find since they manifest themselves through secondary effects or, in the worst case, will not manifest at all prior to your product being shipped. By identifying the source line where the error first occurs, the run-time error checking facilities reduce the time spent in the debugger, and increase the quality of your software. You can specify whether the application will terminate or continue when an error is detected. These optional checks are implemented by generating additional code and/or enabling additional code in the standard C library. Run-time error checking has a nominal effect on code size and execution speed and can be enabled on a module by module basis, making it practical for use in debugging large applications.

The following types of checks are provided:

- **Bounds checking verifies all pointer operations to detect buffer overflows and other illegal operations such as:**
 - accessing uninitialized or null pointers
 - accessing objects outside their declared bounds
 - illegal pointer arithmetic
- **Malloc / free checks uncover dynamic memory allocation errors such as:**
 - buffer overflow
 - write to freed memory
 - multiple calls to free
 - passing an invalid pointer to free
- **Report an unhandled case value in a switch without a default part**
- **Stack overflow detects when the stack grows beyond its allocated size**
- **Divide by zero issues a message when a division by zero is attempted**

MISRA C

Altium was the first company to fully integrate MISRA C support into C compilers for embedded development purposes. MISRA C guides programmers in writing more robust C-code by defining selectable C-usage restriction rules. Through a system of strict error checking, the use of error-prone C-constructs can be prevented. The latest step in this innovation is configurability of the compliancy checking. The MISRA rules, which the application's source code should be compliant with, can be set as 'required' or 'advisory' and the diagnostic level of the generated messages by the compiler can be defined as either 'warning' or 'error'. This allows you to configure the individual rules of the MISRA C compliancy validation according to the quality standards set by your company.

The ARM VX-toolset supports the new MISRA-C:2004 standard as well as the original MISRA-C:1998 guidelines.

CROSSVIEW PRO DEBUGGER

An easy-to-use interface with powerful and extensive debugging features helps you debug your applications faster. CrossView Pro provides multiple, resizable and independently controlled windows, which provide you with all the information required.

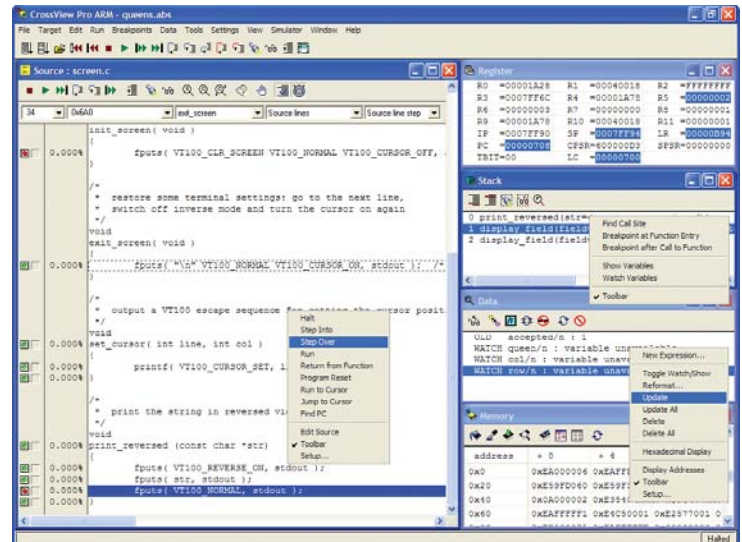
You choose the windows needed to view the different aspects of your code during debugging. It combines the flexibility of the C language with the control of code execution found in assembly language, adding functionality that reduces time spent on testing and debugging.

Functionality includes:

- **Simple through to advanced debugging features**
- **Tracking scope and monitoring locals**
- **Intuitive navigation through the source window**
- **Double-click, right-click and tip-point functions**
- **Bubble-Spy™ for easy inspection of variables and functions**

Source window

The source window is the main debugging window. It allows you to view source; step through your code; set and clear breakpoints, assertions and code coverage markers; watch and show variables; search for strings, functions, lines and addresses; and evaluate expressions. The source window can display code in C/C++ source, assembly or a mixed mode that allows a simultaneous view on your C/C++ source, intermixed with the corresponding assembly code. In order to allow immediate access to your source files, you can jump directly from the CrossView Pro source window into the EDE editor at the exact source line.



Multiple information windows

In addition to the source windows, CrossView Pro offers a wide range of information windows allowing you to navigate through your application, monitor and modify data objects, CPU registers, the stack and memory locations.

The **data window** enables you to watch and modify data objects. Data structures can be shown collapsed as well as expanded. Objects can be displayed in any format on an element-by-element basis. **Register windows** can be configured to display any set of CPU registers and their values. Defining multiple register windows helps you organize your focus. The **stack window** displays the contents of the function-call stack frame. You can easily configure stack-level breakpoints, navigate to the function-call's source and monitor local variables for selected functions. The **memory window** enables you to monitor and modify any memory location with complete control over size and format of the data, as well as view coverage of the memory range.

All information windows are automatically updated, and changed values are highlighted for easy identification.

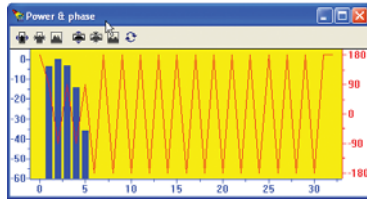
Advanced breakpoints

Breakpoints halt program execution and return control to the user. In addition to industry-standard code and data breakpoints, you can configure your application to halt based upon instruction counts, cycle counts or timer counts. All types of breakpoints can be defined as 'stop-and-go' probe points. Probe points briefly halt and immediately resume execution of the application. During the brief period that the application is halted, only user-specified actions will be performed. Through this mechanism, probe points allow least-intrusive debugging of time-critical applications.

Finally, any number and type of breakpoints can be combined into breakpoint sequences. This allows easy specification of the most complex conditions that need examining.

Program performance analysis

CrossView Pro provides several performance analysis capabilities to help you further optimize your application, as well as shorten your debugging session. The following analysis can be represented in detailed reports or visualized in graphs:



- Code coverage
- Profiling
- Programmable data analysis

Easy debugging of RTOS-based applications

Altium's Kernel-aware Debugging Interface (KDI) defines an open standard interface between CrossView Pro and an RTOS-Aware Debug Module (RADM). The RADM adds CrossView Pro capabilities to read, format and report kernel data structures for any commercial or proprietary RTOS. Our generic RADM for OSEK kernels, which is included in the toolset, is based on the ORTI 2.0 and 2.1 language specification.

The RADM extends CrossView Pro with impressive kernel-aware debugging. Features include:

- Display levels of kernel information
- Examine and modify kernel data structures
- Obtain a summary of all tasks
- View contexts of tasks
- Inspect message contents (pipes, queues, mailboxes)
- Status of synchronization mechanisms
- Interrupt service routine status

CUSTOMER SUPPORT

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

A maintenance period is included with the purchase of TASKING products and entitles you to enhancements and improvements as well as individual response to problems. Annual maintenance agreements are available to prolong this initial support period.

LICENSE MANAGEMENT

The TASKING toolset includes the industry standard FLEXIm license manager, offering stability as well as flexibility. Its license 'borrowing' functionality is a popular feature, allowing laptop users to take a license from the floating license pool for the period of time they are off-site, saving on cost for individual hardware-locked licenses.

PRODUCT PACKAGING & ORDERING CODES

Each TASKING product comes with full documentation in easy-to-use binders. This documentation is also available online as PDF files.

Product code	Package contents
07-200-101-002	EDE, C compiler, assembler/linker, CrossView Pro simulator
07-200-101-012	EDE, C/C++/EC++ compiler, assembler/linker, CrossView Pro simulator debugger
07-200-101-024	EDE, C/C++/EC++ compiler, assembler/linker, CrossView Pro OCDS/JTAG and simulator debugger*

*Product planned, call for availability.

A trial version of the TASKING VX-toolset for ARM is available on CD-ROM or downloadable from our website at

www.altium.com/tasking/ARM

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: tasking.sales.na@altium.com

Asia Pacific

Japan – Altium Japan K.K.

Resona Gotanda Building 7F
1-23-9, Nishi-Gotanda
Shinagawa-ku Tokyo 141-0031
Tel: +81 3 5436 2501
Fax: +81 3 5436 2505
Email: tasking.sales.jp@altium.com

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

Suite A&J, Floor 9, Hua Du Mansion
828-838 Zhang Yang Road
Pudong New Area, Shanghai
Tel: +86 21 6876 4016
Fax: +86 21 6876 4015
Email: info@altium.com.cn

Australia – Altium Limited

Level 3, 12a Rodborough Road
Frenchs Forest NSW 2086
Free Call: 1800 030 949
Ph: +61 2 8986 4400
Fax: +61 2 8986 4440
Email: sales.au@altium.com

Europe

Germany – Altium Europe GmbH

Albert-Nestler-Straße 7
76131 Karlsruhe
Free Call: 0800 0 258486 (0800 0 ALTIUM)
Tel: +49 (0) 721 8244 300
Fax: +49 (0) 721 8244 320
Email: tasking.sales.de@altium.com

Switzerland – Protel AG

(A subsidiary of Altium Limited)
Clarastrasse 12
4058 Basel
Tel: +41 (0) 61 666 68 68
Fax: +41 (0) 61 666 68 69
Email: info.ch@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