

HIGHLIGHTS

- Total integrated Embedded Development Environment
- ISO C++/EC++ compiler
- 16- and 24-bit mode support of DSP563xx
- State-of-the-art debugger interface
- Easy migration from Motorola to TASKING tools
- Motorola CLAS output format
- Comprehensive optimization techniques
- Order utility to create bootable EPROM images
- CrossView Pro support for ADS/EVM
- Available for PC/Windows and Sun/Solaris

INTRODUCTION

The TASKING DSP56xxx Software Development Toolset from Altium is a state-of-the-art programming package for the Motorola DSP56xxx family (DSP566xx, DSP563xx and DSP5600x).

The toolset is now available as version v3.5 and consists of the following:

- ISO C++/EC++ and ANSI C Compiler
- Assembler with Macro Pre-processor
- Linker and Locator
- 'CrossView Pro' debugger with multiple execution environments
 - Instruction Set Simulator
 - ADS/EVM support
- Embedded Development Environment (EDE)

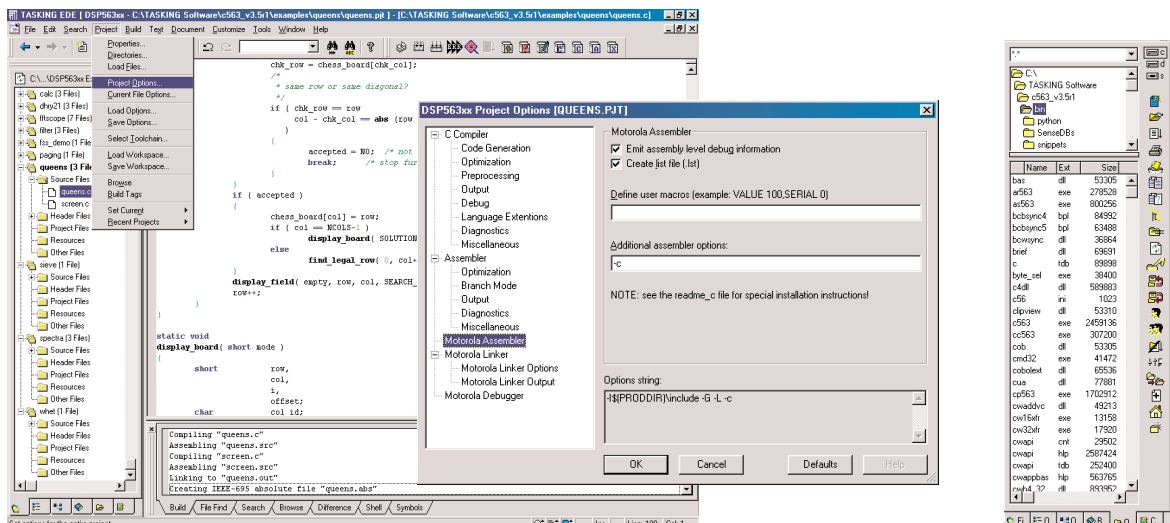
EMBEDDED DEVELOPMENT ENVIRONMENT

With the TASKING DSP56xxx Embedded Development Environment (EDE), you can create and maintain projects easily. All projects 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.

Support for the Motorola toolchain is fully integrated into the EDE so you can easily select the entire TASKING toolchain or the TASKING and Motorola toolchain. To ease migration from the Motorola toolset to the TASKING software development tools, a migration guide is available from www.tasking.com/DSP56xxx.

The DSP56xxx EDE offers many productive features for application and code development, including:

- An explorer-like treeview control allows simplified configuration of the TASKING tools and the DSP target processor for the experienced as well as the novice user
- Menu structure is now tuned according to the development work flow, offering an intuitive project management setup
- Easy selection of target processor and new edit controls for project settings
- Project spaces that enable you to group multiple projects in one view, thus offering improved project management for more complex developments
- CodeFolio to enable easy insertion of template code, thus adding to coding efficiency and consistency. It allows macro expansion and prompted input as you insert the code



The TASKING EDE makes code development and project management easy

C++/EC++/C COMPILER

- Complete compatibility with Motorola GNU compiler
- ISO C++/EC++ and ANSI C compiler
- Static, reentrant, and mixed memory models
- Storage pacifiers for X, Y, L, and P memory
- Optimizing assembler

C++/EC++ COMPILER

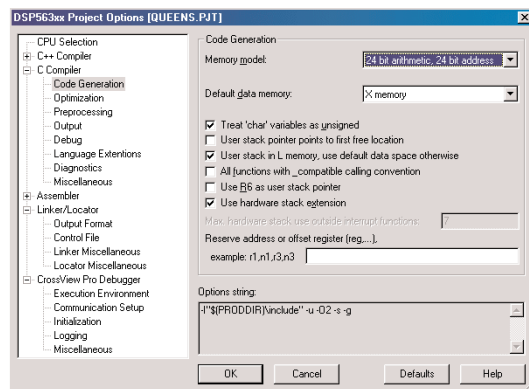
The TASKING DSP56xxx toolset offers the full range of the TASKING C++, C and assembly programming languages. The TASKING ISO C++ compliant compiler delivers the power of object-oriented design and coding techniques to the DSP56xxx family. The object-oriented benefits of C++ can be incorporated into your DSP56xxx application one module at a time, providing appropriate use of assembly, C and C++.

Scalable C++

Fully compatible with the Embedded C++(EC++) standard, the DSP56xxx C++ compiler can be configured to selectively disable C++ features that may not be essential for your embedded DSP application. By selecting (partial) compliance with the EC++ standard, code-size overhead and run-time inefficiency can be minimized.

C COMPILER

Based upon TASKING's DSP C compiler expertise, the DSP56xxx C compiler is powerful, easy to use, and allows you to take full advantage of the DSP56xxx architecture by generating the most optimal code possible. Use of the fractional datatype and memory space qualifiers allows the compiler to optimize loops extensively and exploit the parallel execution capability of the DSP. The compiler supports the 16- and 24-bit mode of the DSP563xx.



Comprehensive view on all project settings

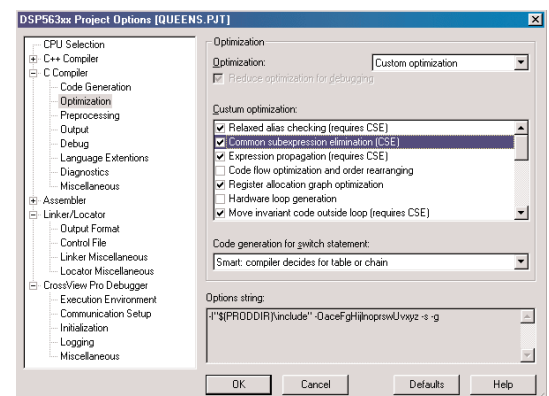
CLAS Compatibility

The TASKING DSP56xxx toolset supports several features that provide compiler interoperability with the Motorola toolchain. By default, the compiler uses a different calling convention than the CLAS compiler. This calling convention guarantees better use of registers, and therefore, less function call overhead. A special keyword (compatible) allows you to prototype individual functions, thus allowing the compiler to apply the more optimal scheme when possible and use the compatible CLAS protocol when necessary.

Compiler Optimization

State-of-the-art optimization techniques are applied to reduce the size of generated code and/or increase execution speed. The following are a sample of the optimization techniques used:

- Compiler generated DO and REP loops
- Effective use of DO-FOREVER and BRKcc
- MAC instruction in computational expressions
- Near, internal and external memory qualifiers
- Fract data type for fixed point arithmetic
- Complex data type
- Built-in support for overflow/saturation
- Circular buffer declarations with `_circ` type qualifier
- Cache handling pragmas
- Subexpression elimination
- Loop recognition
- Variable usage analysis
- Register contents tracking
- Automatic stack overflow checking



Many optimizations possible for both code-size and execution time

In addition to the extensive optimizations, various other features help you to optimize and tune your code:

In-line expansion of predefined functions such as: `_abs`, `_asm`, `_rol`, `_ror`, `_stop`, `_cmac`, `_cmul`, `_cadd`, `_cddiv`, `_nop`, `_swi`, `_wait`, `_round`, `_pdiv`, `_fsqrt`. In-line functions do not incur the typical function call overhead: they translate directly and have no direct equivalent in C.

Adjustable code generation with `#pragma`'s. To control the individual compiler optimizations, allocation of character arrays ('packed' or a character per memory word to optimize for data size or code speed respectively) and `pragma`'s for cache handling.

Circular buffer type modifier for efficient filter implementation.

Floating point libraries with limited exception trapping to reduce runtime argument checking.

Memory models to control the allocation of parameters and automatics to fit the needs of your application. For the DSP5600x, a static model reduces the use of more expensive stack relative addressing modes. The reentrant model provides true stack allocation of objects, and with the mixed model, you can use a mixture of both flavors, tuning your code to use the stack only at those places where you really need it.

Data Types and Sizes

All ANSI C types are supported. Additionally a fixed point data type (*_fract*) has been included.

Data Type	DSP566xx	DSP563xx/00x
	size in bits	size in bits
(un)signed char	8	8
(un)signed short	16	16
(un)signed int	16	24
(un)signed long	32	48
(long)_fract	16(32)	24(48)
pointer	16	16/24
float/double	16+8	24+8
enum	16	24

Libraries

The compilers are delivered with libraries for all the different members of the DSP56xxx families. Each set consists of ANSI C libraries, C++ template libraries, runtime libraries, fixed and floating point libraries. The types float and double are both implemented as single precision floating point.

ASSEMBLER

Features:

- **Compliant with Motorola's CLAS assembler package**
- **Supports nested and fragmented sections**
- **Selects shortest possible branch (forward and backward)**
- **Accepts same model options as compiler**
- **Macro preprocessing**
- **Extended set of controls and pseudo instructions for section handling and list generation**
- **Built-in functions can retrieve the model, default, and stack memory space during code generation**
- **Extensive flow analysis**
- **Parallel instruction execution**

To improve code quality, the assembler performs extensive flow analysis to determine how instructions can be rearranged to reduce the code size and increase execution speed. The assembler also checks whether instruction sequences with pipeline hazards occur and attempts to avoid them by reorganizing the code. If no suitable instruction sequences are available, then the assembler inserts a NOP instruction to force deterministic (and expected) behavior of the program.

The assembler has a built-in macro pre-processor which features an include file mechanism, macro definition and expansion, as well as conditional assembly. Macros can be defined and undefined at any place in the source. The pre-processor supports a set of controls which help you to create structured assembly programs. Move instructions can be executed in parallel with other instructions if no resource conflicts occur. By automatically rearranging instructions such that parallel execution becomes possible, the assembler saves you time while making your code as compact and fast as possible.

For example, the code:

```
move x:Fbuffer_p,r6
gmove #0, b
```

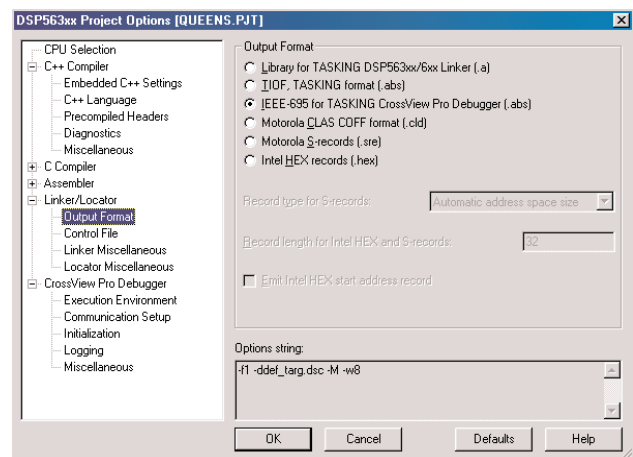
can be replaced by the single instruction:

```
c1r b x:Fbuffer_p,r6
```

LINKER/LOCATOR

Features:

- **Overlaying to reduce memory usage**
- **Partial linking**
- **Incremental linking**
- **Linker and locator map file with symbol values and memory assignments**
- **Flexible locator control language to control memory layout of your application**
- **Locator control files supplied for most commercial targets**
- **Automatic inclusion of library modules**
- **Global type checking**
- **IEEE-695, Motorola S-records, Intel-hex and CLAS (without symbolic information) locator output formats**



Select the object output format desired

Order Utility

Bootable EPROM images can be created by the Order utility, which is integrated in the compiler. Data will be extracted from the .abs file, which will be used to create images for programming a single 8-bit or parallel 24-bit eprom.

CROSSVIEW PRO DEBUGGER

The TASKING DSP56xxx CrossView Pro debugger is a perfect partner in checking, verifying and debugging your DSP application. With its easy-to-use interface and powerful, extensive debugging features, CrossView Pro helps you debug your application faster. CrossView Pro provides multiple, resizable, and independently controlled windows. You choose the windows you need to view the relevant aspects of your code during debugging. 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 on testing and debugging.

Functionality include:

- Simple as well as advanced debugging features
- Intuitive source window
- Bubble-Spy™ for easy inspection of variables and functions
- Double-click and right mouse button functions
- Clipboard copy and paste

Source Window

The source window is the main debugging window. It allows you to view source; step through your application; set and clear breakpoints and assertions; watch and show variables; lines and addresses; call functions; evaluate expressions; and view performance analysis data. The source window can display code in C++/C source, assembly, or a mixed mode that allows a simultaneous view of your C++/C source intermixed with the corresponding assembly code.

Multiple Information Windows

CrossView Pro offers a wealth of information windows that allow you to navigate through your application and monitor and modify Data objects, CPU registers, memory locations and the function-call stack.

The data window enables you to watch or show data, browse for locals or globals, double-click to modify values or to expand and contract complex data structures. Within this window, you can reformat (change display of 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 variables as show or watch.

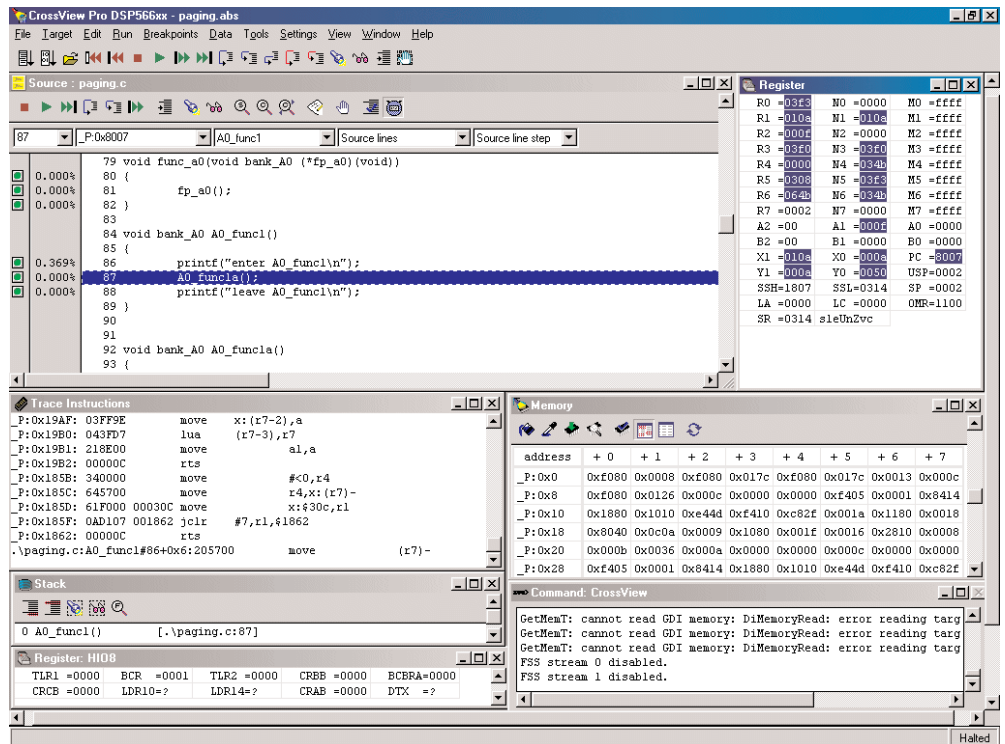
Register windows allow display and modification of CPU register values. Register windows are fully configurable to display any set of CPU registers. By defining multiple register windows you can easily organize your focus.

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.

Advanced Breakpoints

Breakpoints halt program execution and return control to the user. In addition to industry standard memory 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



Spent less time debugging with CrossView Pro

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 time critical applications to be debugged as unintrusively as possible. 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.

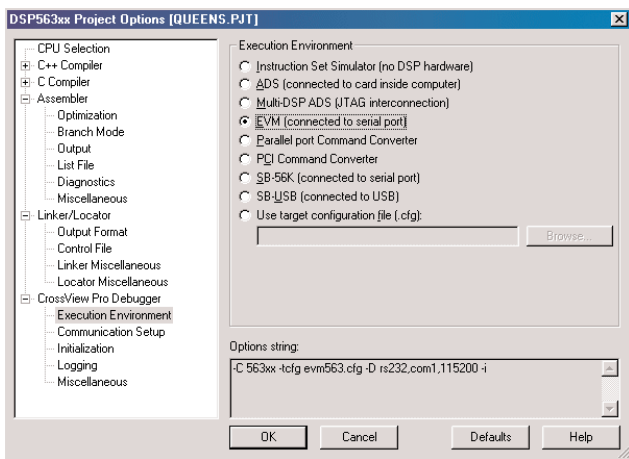
Multiple Execution Environments

DSP56xxx Instruction Set Simulator

With CrossView Pro and the bundled DSP56xxx Instruction Set Simulator, you can debug your application on the host platform even before your target hardware is available. The simulator supports all instructions of the DSP56xxx Instruction Set.

Target debugging via ADS/EVM

The TASKING DSP56xxx CrossView Pro debugger interfaces to the Motorola Application Development System (ADS) and the DSP56xxx Evaluation Modules (EVM).



Easy selection of Execution environment

Program Performance Analysis

The TASKING CrossView Pro debugger provides several performance analysis capabilities to help you further optimize your application as well as shorten your debugging session.

These capabilities include:

- Profiling
- Code Coverage
- Programmable Graphical Data Analysis

Profiling helps you identify bottlenecks in your code by enabling you to perform timing analysis by providing timing information about a particular function or set of functions. You can see how often a function is called and how much time is spent in each function.

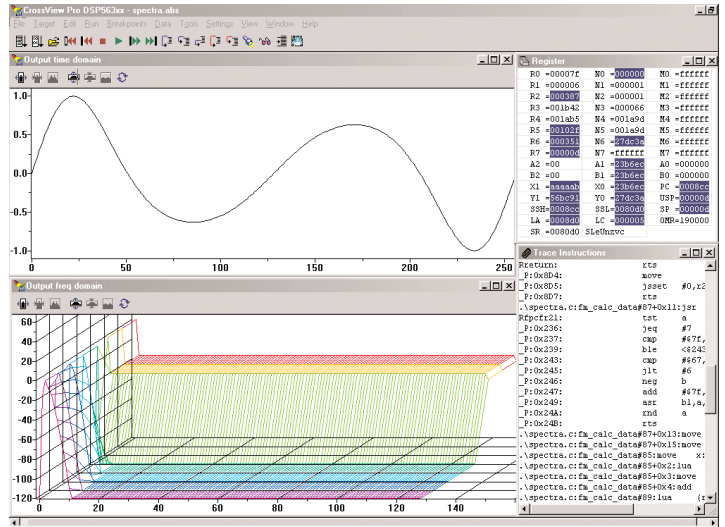
The **programmable graphical data analysis** feature reduces large sets of data into meaningful visual diagrams to enable quick detection of gross errors in the input signal. CrossView Pro analyzes the data, according to pre-defined or user-defined specifications, and then displays the data the way you need it. You can also view the same data in several ways at the same time (for example, in the time and frequency domain). Five pre-defined analysis types are now available: x-t plotting, x-y plotting, FFT power spectrum, FFT waterfall, and Eye diagram.

Code coverage tracks all memory access (memory read, memory write, instruction fetch) so you can determine if there are any areas of unexecuted code. You can also use one of the timers for cycle counting on the real hardware. Call counts, tim-

ing per line, timing per assembly instruction, and coverage are available in the simulator.

EASY Debugging of RTOS-based Applications

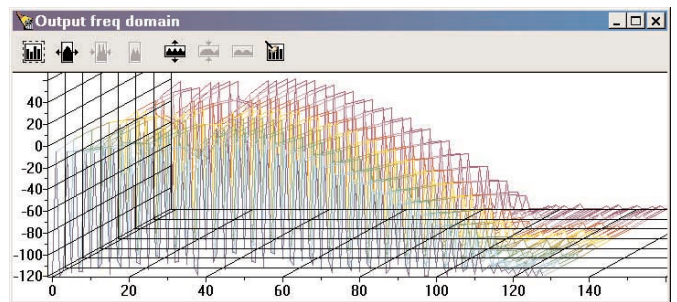
TASKING's Kernel-aware Debugging Interface (KDI) defines an open standard interface between CrossView Pro and an RTOS Aware Debug Module (RADM).



View large sets of data in a meaningful form

Which can be used to add kernel-awareness to CrossView Pro for any commercial or proprietary RTOS. The RADM extends CrossView Pro with impressive Kernel Aware Debugging capabilities, such as:

- 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



Clear visual view on CrossView Pro data

COOPERATION WITH THIRD PARTIES

Altium's extensive third party cooperation ensures that you have access to the tools you need to be your most productive. We work closely together with manufactures of Emulators, RTOS solutions and Design Tools for the DSP56xxx.

CUSTOMER SUPPORT

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 extend this initial support period.

PRODUCT PACKAGING & ORDERING CODES

Each TASKING product comes with full printed documentation. This documentation is also available on-line in the form of a Windows Help system, HTML and PDF and provides full-text search capabilities for quick and easy access to topics.

Combination Package for all DSP566xx, DSP563xx and DSP560xx

Product Code	Package Contents
TK039-024	EDE, C and C++/EC++ Compiler, Assembler/Linker, CrossView Pro Simulator
TK039-049	CrossView Pro debugger ADS/EVM support

Demonstration versions of the TASKING DSP56xxx tools are available on CD-ROM or downloadable from our web site at: www.tasking.com/dsp56xxx

INTERNET

Web site: <http://www.tasking.com>
Discussion forum: <http://groups.yahoo.com/group/TASKINGforum>

DISTRIBUTOR

ALTIUM SALES OFFICES



North America

Altium Inc

17140 Bernardo Center Drive, Suite 100
San Diego, CA 92128
Toll Free: 877-TASKING
Tel: 858-521-4280
Fax: 858-485-4610
E-mail: tasking.sales.na@altium.com

Asia - Pacific

Japan - Altium Japan KK

ASAHI-GIN Gotanda Building 7F
23-9, Nishi-Gotanda 1-chome
Shinagawa-ku Tokyo 141-0031
Tel: 03 5436 2501
Fax: 03 5436 2505
E-mail: tasking.sales.jp@altium.com

Australia - Altium Limited

Level 14, 39 Murray Street
Hobart TAS 7000
Free Call: 1800 030 949
Fax: 03 6231 4167
E-mail: sales.au@altium.com

Europe

Germany - Altium Germany GmbH

Albert-Nestler-Straße 7
76131 Karlsruhe
Free Call: 0800-0-258486 (0800-0-ALTIUM)
Fax: (0)721 82 44 320
E-mail: tasking.sales.de@altium.com

Switzerland - Protel AG

(A subsidiary of Altium Limited)
Unterdorfstrasse 1
CH-4334 Sisseln
Tel: (0)62 866 41 11
Fax: (0)62 866 41 10
E-mail: tasking.sales.ch@altium.com

From Austria

Free Call: 00800 776 776 77
Free Fax: 00800 776 776 00

From France

Free Call: 0800 88 05 06
Free Fax: 0800 82 85 92

European Free Call Numbers

German speaking: 00800 776 776 77
French speaking: 00800 776 776 55
English speaking: 00800 776 776 44
Free Fax: 00800 776 776 00