

## HIGHLIGHTS

- Integrated tools delivering a rapid edit-compile-debug process
- Easy project set up and management
- Customize to your own environment
- C/C++/EC++ compiler
- Comprehensive optimization techniques
- Basic and advanced debugging
- Program performance analysis in debugger
- Open architecture and industry standards
- RTOS-aware

## THE RIGHT DEVELOPMENT SOLUTION

You have a tight deadline and need to be able to spend your time developing and testing your application. To do this, you require tools that not only help you be more productive, but also help you produce the most efficient code. The TASKING PowerPC (PPC) Software Development Toolset helps you produce highly optimized code, saves valuable time, and gets your product to market faster.

The TASKING PPC Software Development Toolset:

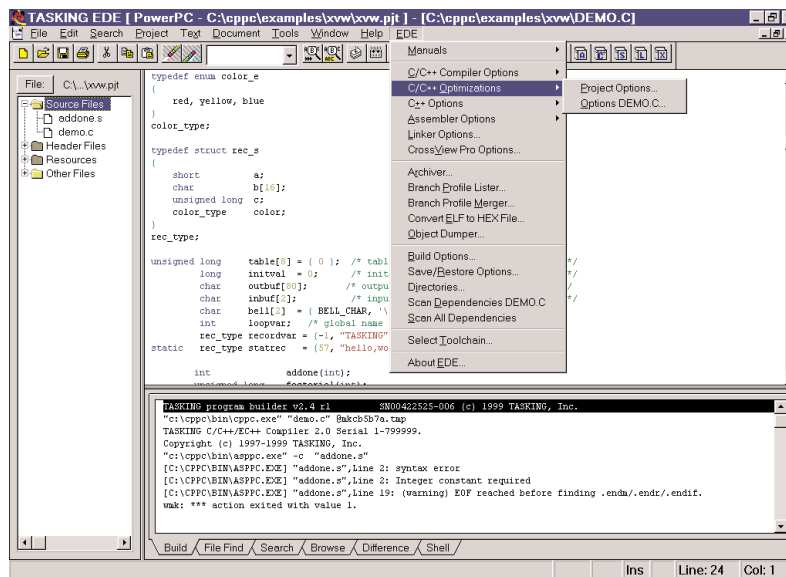
- Integrates all the tools you need for embedded application development into a single user interface for rapid and easy navigation
- Simplifies project management by enabling you to setup your project environment only once and/or define multiple project environments
- Expedites testing via file system simulation and recording/playback of debug sessions
- Helps produce highly optimized code via compiler optimizations, integrated profiling (time and branch), code coverage and programmable graphical data analysis
- Provides extensive yet easy to use debugging capabilities for simple and elusive bugs
- Contains an extensive range of libraries including low level, I/O and memory management functions

The PPC toolset consists of EDE (Embedded Development Environment), C/C++/EC++ Compiler tools and a CrossView Pro debugger.

## EMBEDDED DEVELOPMENT ENVIRONMENT

EDE, voted by EE Times as "Best Technology", supplies a single easy to use interface for all the tools needed to build, edit, compile, and debug embedded applications. EDE also includes the following components:

- Language-sensitive editor
- Automated make facility
- Librarian
- Built in grep and file difference
- Tool option selection
- Object code reporter



# C/C++/EC++ COMPILER

- Specific embedded functionality
- Categories of optimizations
- Embedded C++ (EC++) support
- Branch and time profiling
- Extensive range of libraries
- ELF converter
- Multiple locator output formats

Using the EDE menus, you control the Compiler, Assembler, and Linker-Locator with a simple point and click. The language-sensitive editor understands the error messages generated by the compiler or assembler and shows you where the errors exist so you can fix them quickly. Object modules can be converted from ELF format into hexadecimal so your program can be downloaded into EPROM or flash.

You can define project environments (compile, assemble, make, debug, etc.) with a set of menus and dialog boxes so the files associated with your project will automatically use the PPC suite of tools. File dependencies as well as the exact sequence of operations required to build your application are also handled by EDE. An easy version control interface enables you to check out a file for review, check in your changes, or lock a revision you plan to change.

## THE C/C++ COMPILER

The PPC compiler applies powerful optimization algorithms to produce the most efficient code for your application. The compiler conforms to ANSI C standard X3.159-1989 and the ANSI X3J16C++ standard. The C++ compiler supports templates, dynamic casts, runtime type identification and exception handling. Embedded C++ (EC++) support is available to reduce the high overhead often introduced by C++. The evolving EC++ standard addresses this issue by omitting a number of features that are not essential for most embedded applica-

tions. The EC++ option helps you conform to the EC++ standard. The linker/locator reads and writes files in relocatable or absolute Executable and Linking Format (ELF).

## Compiler Optimizations

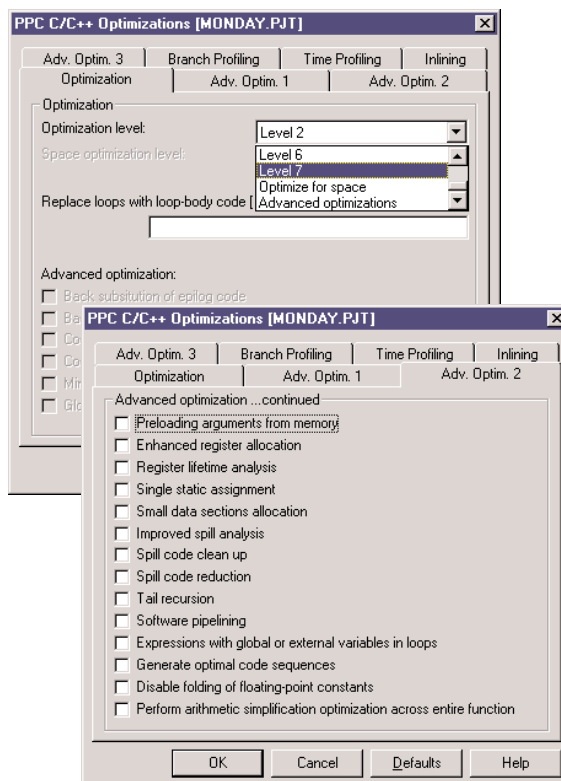
The PPC compiler includes optimizations which provide you with a high level of control for generating the code you require. You control the level of optimization your application needs by selecting predefined levels of optimizations (that address faster compilation, faster code, smaller code) or specific optimizations.

Additionally, the PPC compiler provides you with categories of optimizations that affect:

- The program being compiled
- Only certain portions of your code
- Linkage to external functions
- Data storage
- Memory models

The following is a partial list of optimizations:

- Optimal use of instruction queue and cache by using branch prediction data from previous program execution
- Software pipelining for completely removing instruction dependencies in loops
- User control of function inlining
- Multi-module function inlining
- Literals are kept in code space
- Function arguments can be preloaded to registers
- Storage of string constants with read-only data
- Loop operator strength reductions
- Elimination of unused function definitions across modules
- Branch profile listing of execution instances by source line
- Integrated execution time profiling
- Support for multi-threaded applications
- Global and local iterative common sub-expression elimination



- Production of position-independent code
- ROMable code and constant data
- Flexible asm macros for inlining assembly in C/C++ code
- Instruction scheduling based upon the selected PowerPC derivative
- Optimizing LSWI and STWI load and store string instruction usage
- Optimized function prolog and code
- Aggressive usage of near addressing mode for accessing global data
- Optimizer recognition and optimization of ANSI transcendental functions
- Decrement and branch loop rewriting
- Inlined struct assignments and memcpy()
- Endian byte order reversals as load/store reversed and intrinsic byte swap functions
- Condition code simplified using instructions with record mode
- Interrupt Functions in C
- Floating point FSEL conditional moves
- Floating point transcendental constant folding and strength reduction

### Profiling

The TASKING PPC compiler supports two types of profiling: time and branch. Time profiling is invoked by simply checking this compiler option via EDE and the profile data is automatically copied from the target to your host by CrossView Pro. It enables you to perform timing analysis by recording timing information about a function or set of functions. You can see how often a function is called and how much time is spent in each function.

Branch profiling uses feedback from previous executions of the application to predict what branches will and will not be taken. If the traversal of a branch is correctly predicted, the PPC processor can pre-fetch instructions from the correct target. This reduces pipeline stalls and can save many cycles of processor time.

### THE ASSEMBLER

The PPC assembler is EABI compliant with over 70 assembler directives which enable you to control program organization and manipulate data. The assembler includes a macro pre-processor which features an include file mechanism, macro definition and expansion.

The assembler supports:

- **Macro preprocessing**
- **Structured and conditional assembly**
- **Regular, local and numeric labels**
- **Integer, floating-point, and string constants**
- **An unlimited number of relocatable, absolute, and combinable segments**
- **Built-in symbols which can be used as operands in assembly statements or macro definitions**

### Archiver

The archiver utility groups independently developed object files into an archive library to be accessed by the linker. When the linker reads an archive file, it extracts only those object files necessary to resolve external references.

The archiver performs the following:

- **Creates an archive library**
- **Deletes, adds or replaces one or more members in an archive library**
- **Extracts one or more members from an archive library**
- **Lists the members included in an archive library**
- **Maintains a list of externally-defined names and the associated archive member**

### Libraries and Classes

The PowerPC compiler contains an extensive range of libraries which save valuable development time by eliminating the need to write your own low-level I/O functions and start-up code. The complete source code of all runtime libraries is provided so you can tailor the libraries to your specific needs.

The following fully reentrant libraries are included:

- **ANSI Standard C with extensions**
- **Floating Point including IEEE-754**
- **C++ classes and functions (I/O streams)**
- **Low-level I/O libraries for CrossView Pro (I/O via RAM disk or file system simulation)**
- **Symbolic Register Access**

# CROSSVIEW PRO

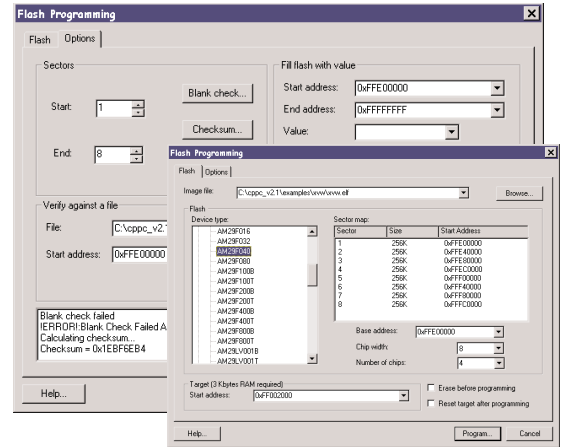
- Graphical User Interface to all features
- Mixed-mode source/assembly display
- Multiple window displays
- Powerful breakpoint control
- Bubble-Spy™ technology for easy and quick inspection of variable contents
- Programmable Graphical Data Analysis
- Code coverage
- Flash programming
- Optimal support of on-chip breakpoint
- File system simulation
- Multiple execution environments
- Debug applications in ROM
- Direct memory and symbolic register access
- Register grouping
- Symbolic register editing
- Record and playback
- C level trace

## CROSSVIEW PRO DEBUGGER

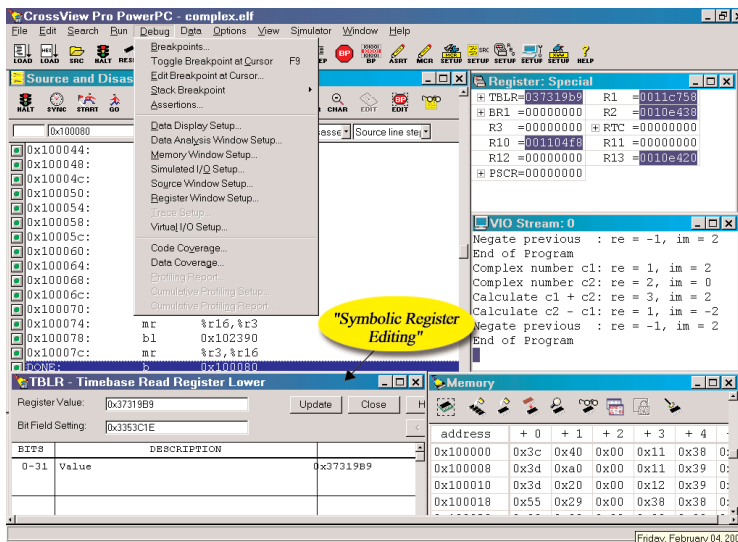
An easy-to-use Graphical User Interface combined with powerful debugging features helps you debug your applications faster. All features are accessible via the GUI so you don't need to enter any text or learn any command strings.

Features include the following:

- Multiple viewing windows (source, register, trace, memory, stack, data, command) display the type and level of information needed at any point during the debugging session
- Display your program as C source text, assembly source or a mixture of both
- Single-step through C or assembly source
- Set, clear, enable, disable breakpoints
- Code, data, complex, stack and hardware breakpoints
- Full support of PowerPC on-chip breakpoint unit provides the capability to debug applications in ROM
- Start and stop recording debug session at any time
- Record and playback debug session to automate debugging or test application
- Double-click to modify values or expand and contract complex data structures
- Integrated Flash Programming capabilities allow in-circuit programming of Flash devices on the target board
- Use assertions for hard-to-find errors



- Watch or show data (global or local)
- Immediately view the values of a variable or function by simply moving your mouse (Bubble-Spy)
- Edit, display and group registers
- Display multiple windows of different register groups
- Symbolic register editing enters the register value by specifying the meaning of the bits in the register and configures Special Function Registers
- Observe the state of current stack frame, including function parameters
- Monitor and edit the current value of memory locations
- Open multiple memory windows for different ranges, and can display a different format (ASCII or numeric) for each window
- Multiple execution environments (simulation, BDM/JTAG, ROM monitor, in-circuit emulator)
- Simulator includes a Peripheral Debug Interface (PDI) enabling you to simulate the behavior of your own peripherals
- File system simulation enables the use of regular I/O functions such as `fopen()` and `fprintf()` to use files on the host system, with both keyboard and screen I/O redirected to CrossView Pro windows
- Personalize your windows to your environment by removing toolbars, changing the buttons on a toolbar or combining a series of CrossView Pro debugger commands assigning them to a button



# BREAKPOINTS AND PROGRAM PERFORMANCE ANALYSIS

## Powerful Breakpoints

Setting breakpoints is the feature most often used in a debugging session. The CrossView Pro debugger provides you with a variety of breakpoint capabilities enabling you to resolve a simple or difficult problem quickly.

- Code breakpoints halt the program at a particular statement or instruction so the values of variables can be observed.
- Data breakpoints let you determine when memory addresses are read and/or written to, and are useful for tracking the possible misuse of pointers, global variables and memory mapped I/O ports.
- Complex breakpoints, after reaching an address, check either a register or memory location for specified values before taking the breakpoint.
- On-chip hardware breakpoints enable you to set breakpoints on any type of memory access or memory range after a number of accesses, and to place breakpoints in ROM.
- Stack breakpoints can be set at either function entry or exit
- An assertion is a command, or series of commands, executed after every line of source code. Assertions can be used to test for a wide array of error conditions throughout the entire length of a program.

## Program Performance Analysis

The CrossView Pro debugger provides performance analysis capabilities to help you further optimize your application as well as shorten your debugging session. These capabilities include Code Coverage and Programmable Graphical Data Analysis.

The profiling capabilities in the compiler and the code coverage functionality available in the CrossView Pro Simulator help you produce more efficient and reliable code. Coverage enables you to trace all memory access (memory read, memory write, instruction fetch) so you can determine if there are any areas of unexecuted code.

The Graphical Data Analysis feature reduces large sets of data into meaningful visual diagrams to enable quick detection of gross errors in the input or output. 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. You can easily define the source data position (memory type and address), source data length, basic type of source data (int, \_fract, float), data processing method (time-domain, power spectrum, power and phase) and data display method (graph, dotsk, bars).

You can also view the same set of data in several ways at the same time, for example in the time and the frequency

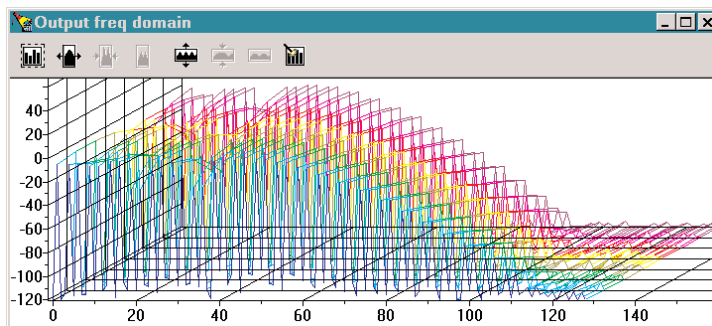
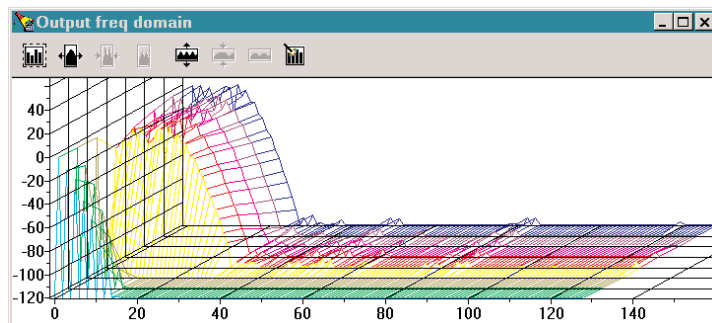
domain. Five pre-defined analysis types are available: FFT power spectrum, FFT waterfall, x-t plotting, x-y plotting, and eye diagram.

## Open Architecture

CrossView Pro 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 framework. The KDI can also be used to provide kernel aware debugging for "in-house" kernels.

## SUPPORTED TARGETS

- PLX Technology IOP 480
- 400 series : 403 (GA, GB, GC)
- 500 series : 505, 509, 555
- 600 series : 603(E), 604(E)
- 700 series : 740, 750
- 800 series : 821, 823, 850, 860, 8240, 8260



## AVAILABILITY

The PowerPC toolset is available for PC/Windows and Sun/Solaris.

## PRODUCT PACKAGING & ORDERING CODES

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

Product Code	Package Contents
TK080-34x	EDE, C/C++ Compiler Toolset, CrossView Pro Debugger (SIM, BDM, ROM or ICE)
TK080-012	EDE, C/C++ Compiler Toolset
TK080-041	CrossView Pro ROM Monitor Debugger
TK080-043	CrossView Pro Extended Simulator Debugger
TK080-044	BDM/JTAG for PowerPC
TK080-046	CrossView Pro Debugger for ICE

\* This is a combination package that includes one EDE, C/C++ compiler toolset and one CrossView Pro debugger execution environment. The "x" indicates the debugger execution environment, for example: -341 for a ROM Monitor, -343 for a Simulator, -344 for BDM/JTAG, -346 for ICE.

## INTERNET

Web site : [www.tasking.com](http://www.tasking.com)  
Developers forum : [www.tasking.com/forum](http://www.tasking.com/forum)

## DISTRIBUTOR

TASKING, the TASKING logo, The Embedded Communications Company and Bubble Spy are trademarks of TASKING. All other trademarks and logos are trademarks or registered trademarks of their respective owners. TASKING retains the right to make changes to the specification at any time, without notice. Contact your local sales office to obtain the latest information. TASKING assumes no responsibility for any errors that may appear in this document.

## UNITED STATES (International Headquarters)

TASKING, Inc.  
333 Elm Street  
Dedham, MA 02026-4530 USA  
Phone: + 1-781-320-9400 (outside USA & Canada)  
1-800-458-8276 (East Coast)  
1-877-TASKING (West Coast)  
Fax: +1-781-320-9212  
Email: [sales.us@tasking.com](mailto:sales.us@tasking.com)

## THE NETHERLANDS

TASKING Software BV  
Plotterweg 31  
3821 BB Amersfoort, The Netherlands  
Phone: +31-33-4558584  
Fax: +31-33-4550033  
Email: [sales.nl@tasking.com](mailto:sales.nl@tasking.com)

## GERMANY

TASKING GmbH  
Eltinger Strasse 61  
71229 Leonberg, Germany  
Phone: +49-7152-97991-0  
Fax: +49-7152-97991-20  
Email: [info.de@tasking.com](mailto:info.de@tasking.com)

## ITALY

TASKING S.r.l.  
Via Napo Torriani 29  
20124 Milano, Italy  
Phone: +39-02-6698-2207  
Fax: +39-02-6698-2189  
Email: [sales.it@tasking.com](mailto:sales.it@tasking.com)

## JAPAN

Nihon TASKING K.K.  
Shiba-ST Building, 6th Floor  
1-15-13 Shiba  
Minato-ku  
Tokyo 105-0014, Japan  
Phone: +81-3-3457-6831  
Fax: +81-3-3457-6834  
Email: [sales.jp@tasking.com](mailto:sales.jp@tasking.com)

## UNITED KINGDOM

TASKING Ltd.  
St. James Road  
Brackley  
Northants NN13 7XY, U.K.  
Phone: +44-1280-706686  
Fax: +44-1280-700577  
Email: [sales.uk@tasking.com](mailto:sales.uk@tasking.com)