µVision3 IDE for Microcontrollers

KEIL SOFTWARE

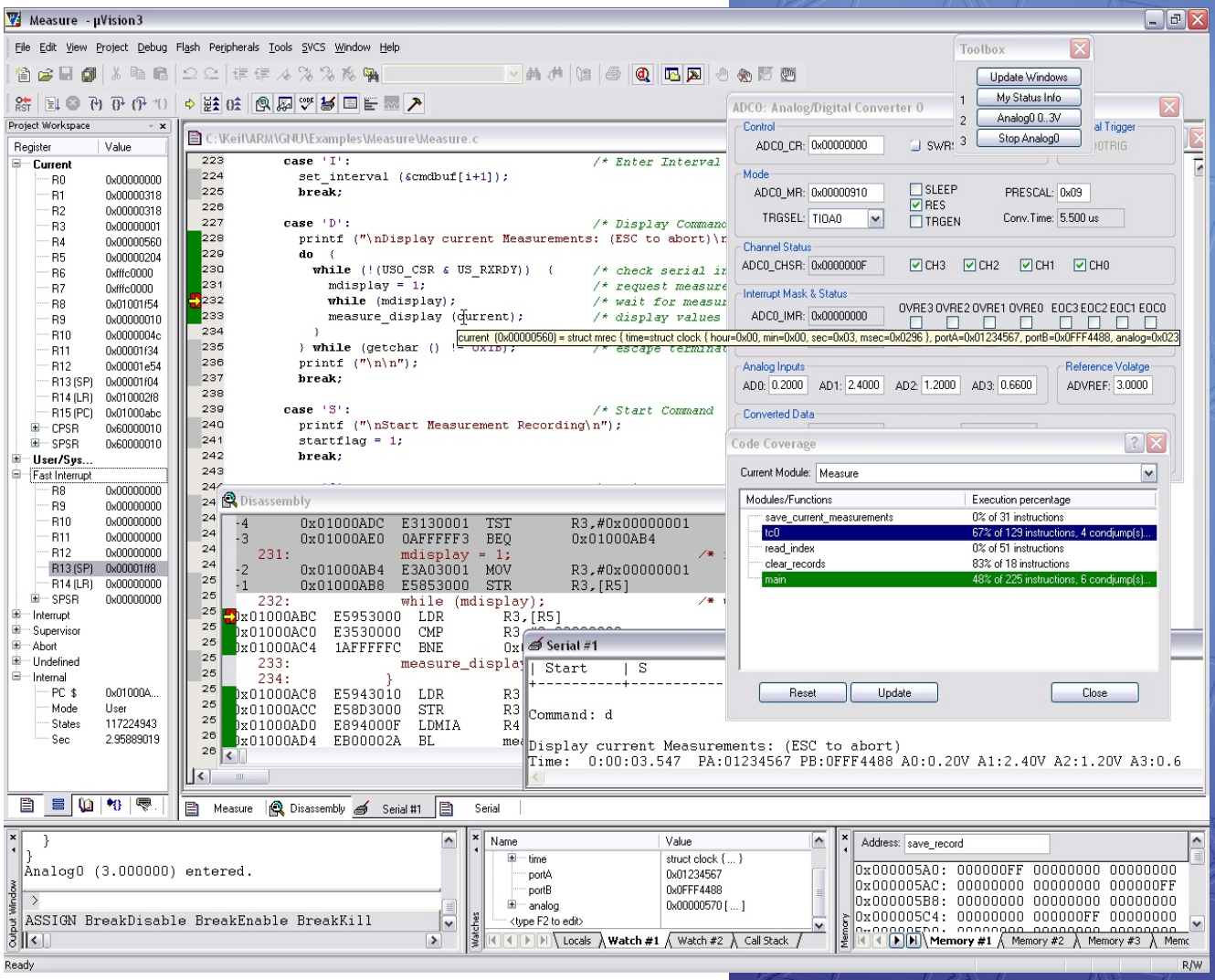
µVISION 3

Quick Start

µVision, the popular IDE from Keil Software, combines Project Management, Source Code Editing, Program Debugging, and Flash Programming in a single, powerful environment. This Quick Start guide gives you the information necessary to use µVision3 for your own projects. It provides an overview of the most commonly used µVision3 features including:

- Project Management, Device Setup, and Tool Configuration.
- Editor facilities for Creating, Modifying, and Correcting Programs.
- Target Debugging or CPU & Peripheral Simulation.

For experienced users, µVision3 adds new features such as Source Outlining, Function Navigation, Editor Templates, Incremental Search, Configuration Wizard, Logic Analyzer, CAN and I^2C Simulation, Flash Programming, and JTAG Debugging.



The Keil µVision3 IDE supports over 700 devices including ARM-powered, XC16x, C16x, ST10, 251, and 8051-based microcontrollers.

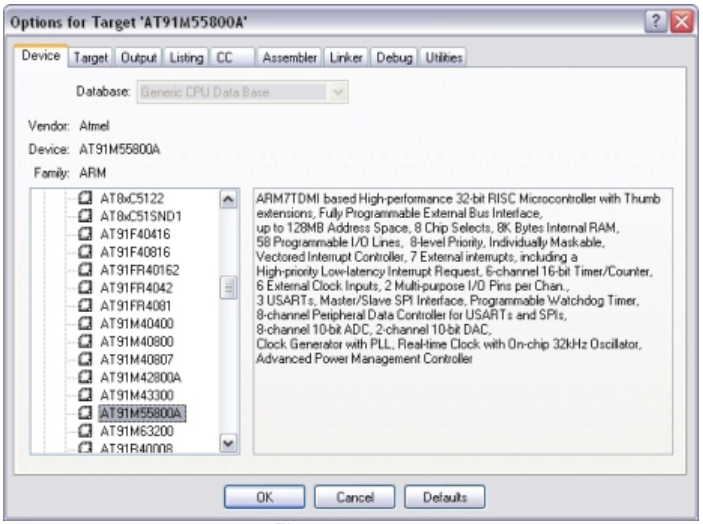
www.keil.com

Project Management

µVision3 ensures easy and consistent Project Management. A single project file stores source file names and saves configuration information for Compiler, Assembler, Linker, Debugger, Flash Loader, and other utilities. The **Project** menu provides access to project files and dialogs for project management.

Open Project... loads existing project files. The **Examples** folder contains many useful sample applications like the **Measure** example that is used throughout this Quick Start guide.

New Project... allows you to create your own projects.

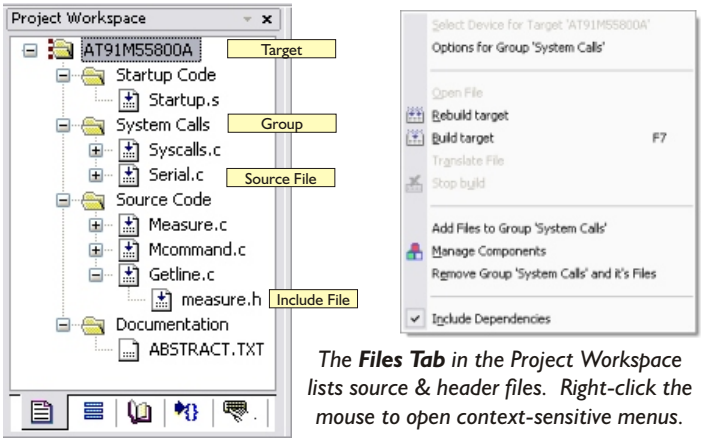


The Device Database™ simplifies tool configuration dialogs and provides the basic setup for the development environment.

Device Database™

When you start a new µVision3 project, you select the target device from the Device Database™ and µVision3 automatically sets required tool options and customizes dialogs.

µVision3 displays only those options that are relevant to the selected device and prevents selection of incompatible directives. Just a few dialogs are required to completely configure all the tools (Assembler, Compiler, Linker, Debugger, and Flash Download Utilities).

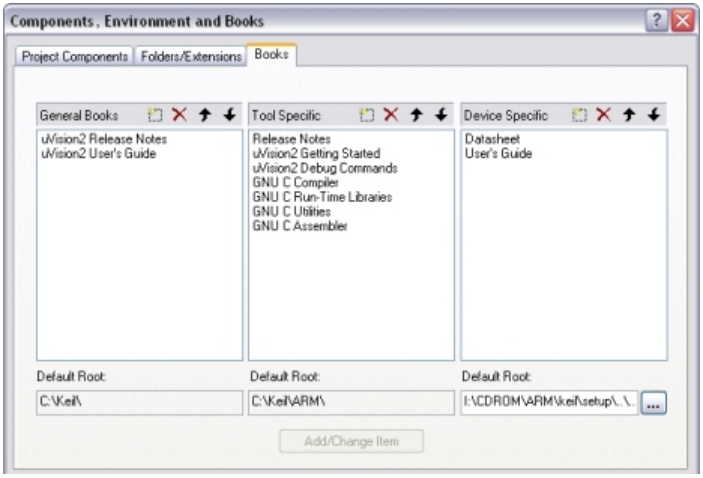


*The **Files Tab** in the Project Workspace lists source & header files. Right-click the mouse to open context-sensitive menus.*

Project Components

File Groups allow you to group associated files together. They may be used to separate files into functional blocks or to identify engineers in your software team.

Project Targets allow you to create several programs from a single project. You may require one target for testing and another target for a release version of your application. Each target allows individual tool settings within the same project file.

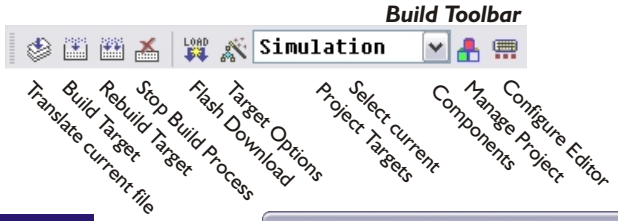


The Manage Components dialog allows you to create File Groups and Project Targets and add documents (Books) to a project.

User's Guides & Additional Books

The **Books Tab** in the Project Workspace lists extensive on-line manuals for the tools and the selected microcontroller. You may add your own additional books using the **Manage Components** dialog. This provides quick access to CPU and project documentation.

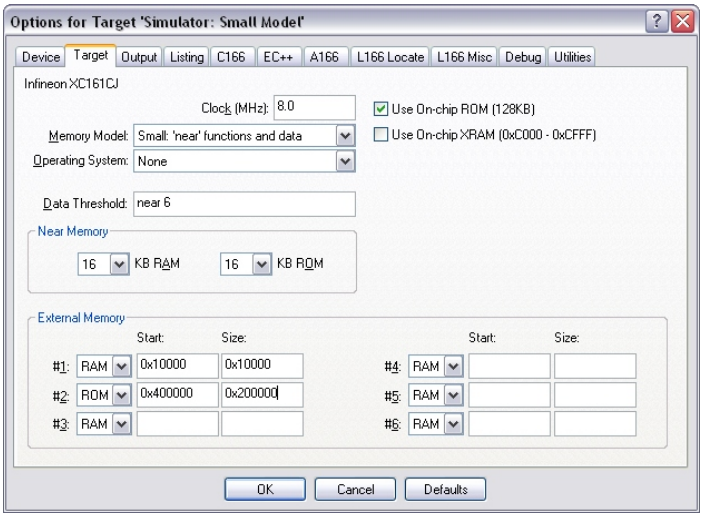
Tool Configuration & Make



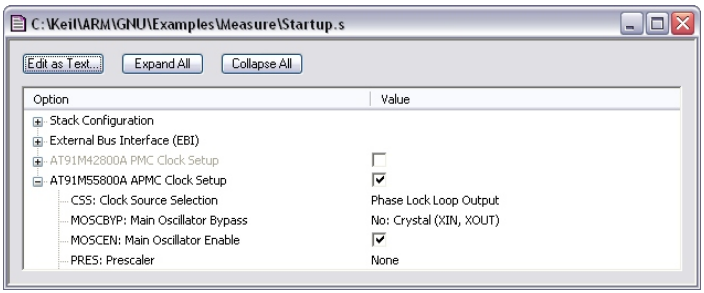
Development Tool Options

µVision3 allows you to set tool options by target, group, and source file. The options dialog opens via the context menu of the Project Workspace and provides tabs for:

- **Device**: Allows you to view and change the selected target CPU.
- **Target**: Allows you to specify CPU and memory options. These are used to configure basic tool settings including those of the linker, debugger, and simulator. Typically, these are the only settings you must make to begin a new project.
- **Output/Listing**: Allow you to configure the generated object and listing files.
- **Compiler/Assembler/Linker**: Allow you to view and change tool-specific options.
- **Debug**: Allows you to configure the µVision3 Simulator and Debugger.
- **Utilities**: Allows you to configure project-specific tools (like Flash programming).



The Device Database™ pre-configures tool settings for the selected CPU. Only few dialogs pages are required for complete tool setup.



The Configuration Wizard provides a graphical editor for startup files, configuration code, debugger INI files, and linker scripts.

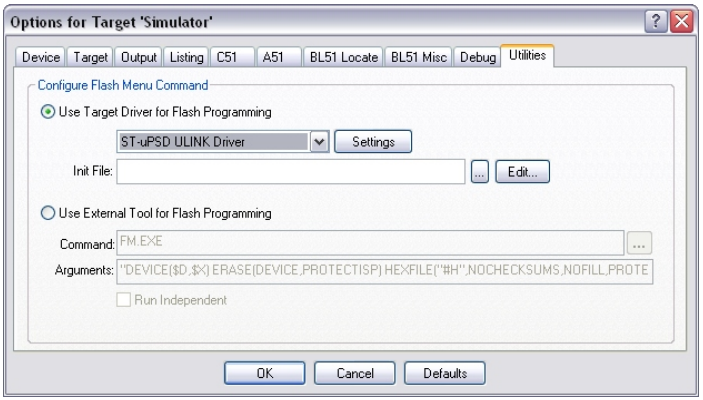
Build Process

The **Build Target** command runs the Compiler and Assembler. The tools automatically generate file dependencies so only those files that have changed are re-translated. You may enabled additional Global Code Optimizations which are performed by incremental re-compilations of C modules.

Flash Download

Flash Download features are included in some Target Drivers (like the **EPM900** LPC Emulator/Programmer and the Keil **ULINK** USB JTAG-Adapter).

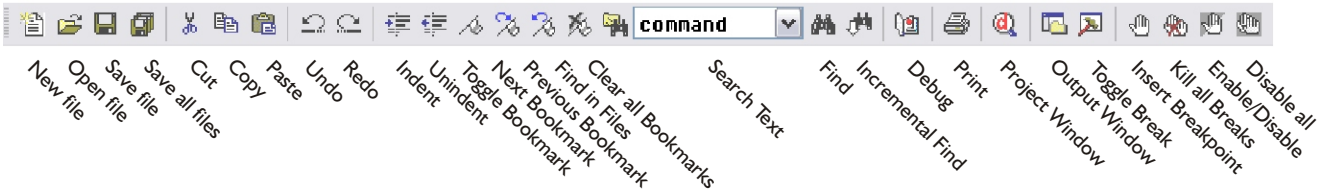
Additionally, you may integrate external Flash utilities (and their command-line options) into the µVision3 project setup.



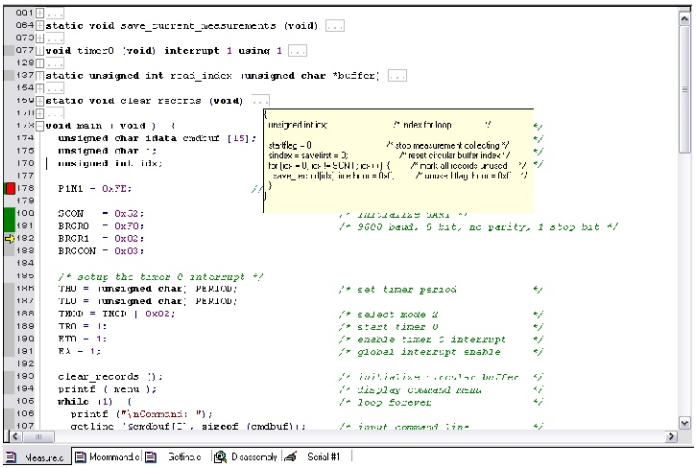
The Utilities tab adds Flash Download Utilities configuration to the project.

Source Code Editor

Editor Toolbar



The µVision3 editor offers many standard and advanced software editing features. Color syntax highlighting and text indentation are optimized for editing C source code. Most editor functions may be quickly accessed from the toolbar or the context menu. Program debugging is directly controlled in the editor. This natural environment allows you to quickly integrate and test source code changes.



µVision3 includes all standard editor features. Advanced Features provide an overview of source files and help you complete your task faster.

Advanced Features

µVision3 adds many new editor features like:

- **Incremental Find** positions the cursor while you type the search phrase.
- **Active Brace Checking** shows nesting and highlights mismatches while entering parentheses, braces, or brackets.
- **Text Block Functions** that Comment, Indent, Uppercase, Tabify, and Remove Whitespace from text blocks.
- **Document-Selective Settings** for tab spacing and syntax coloring in Assembler, C, and other file types.
- Detailed **Syntax Highlighting** that allows you to define a user keyword list. Colors are used in printed output.
- Document **Outlining** that provides a quick overview of complex source files.

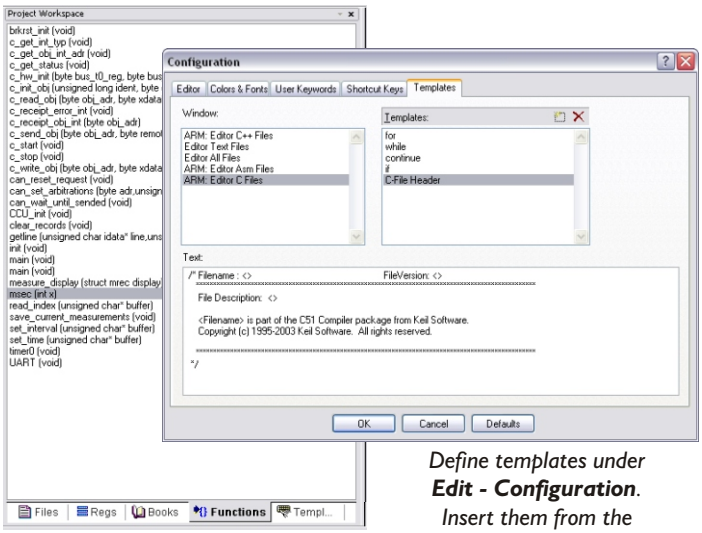
```
if (arr[1])
if (arr[1]
```

***Active Brace Checking** provides an overview of complex statements. When editing source files, matching parentheses, braces, or brackets are marked in grey—mismatches are shown in red.*

Functions and Templates

The **Functions Tab** in the Project Workspace displays all functions in either your project or open editor files. You may click on a function to jump to its definition.

The **Templates Tab** shows user-defined text blocks (templates). To insert a template into your source file, type the first few letters of the name followed by **Ctrl+Space** or click the item in the Templates Tab.



*The **Function Tab** alphabetically lists functions in your project. Click to jump to the function definition.*

*Define templates under **Edit - Configuration**. Insert them from the **Templates Tab** or by entering the first few letters of the name followed by **Ctrl+Space**.*

Editor Utilities

µVision3 contains powerful features and interfaces to external tools which help you complete your projects on time.

Source Browser

The integrated Source Browser (available for Keil Cx51, C251, and C166) maintains a database of your program symbols for quick navigation through your source files. You may use the detailed information to optimize the memory types of variables.

PC-Lint Syntax Checking

Gimpel Software's PC-Lint integrates into the µVision3 IDE and is independent of the Compiler. PC-Lint checks your C/C++ source code for bugs, glitches, inconsistencies, and non-portable constructs. PC-Lint Version 8 also includes checking for MISRA rules.

Version Control Systems

A configurable interface allows you to plug-in the Software Version Control System (SVCS) of your choice. Pre-configured templates are provided for: Intersolv PVCS, Microsoft SourceSafe, MKS Source Integrity, Rational Clear Case, Perforce, and CVS (available under the GNU open source license).

Remote Control Interface

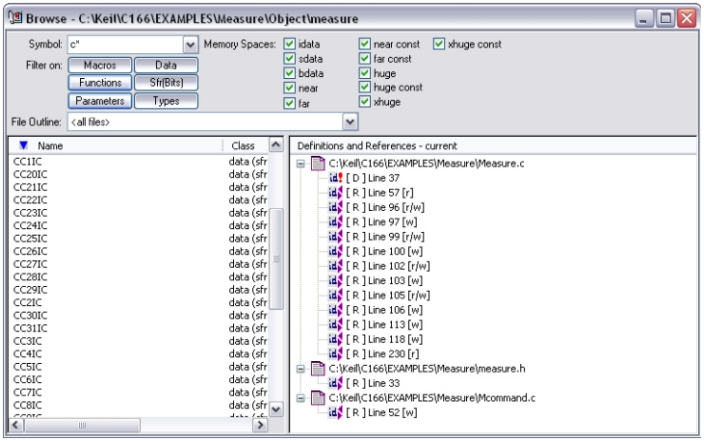
µVision3 may be controlled via the Development Tool Connection (DTC) Interface. DTC is used by third party tools like the iLogix/WST UML Bridge, BKR EasyCode, or Hitex Tessy to remotely control IDE and Debugger functions.

Find in Files

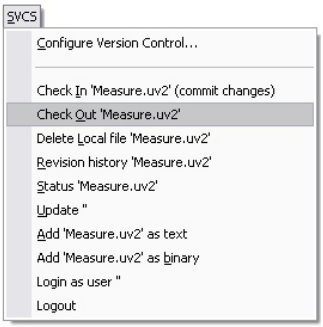
Find in Files performs a text search in all specified files. The search results are displayed in the Find in Files page of the Output window.

External Tools

The Tools menu integrates your favorite utility programs into µVision3. Project-related arguments (like file names or project folders) may be sent as arguments to these utilities.



Use the Source Browser for fast navigation to definitions and references.



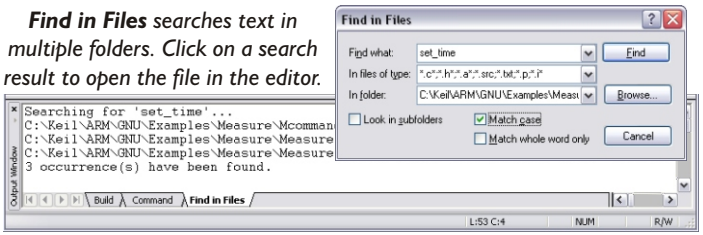
Project Workspace Icons

- Standard Source File
- File or File Group with specific Options
- File checked in SVCS (R/O)
- File excluded from linkage

Key Sequences are used for argument passing to external tools, like SVCS. Prefixes (%, $, @, ...) specify which part of the path to use.

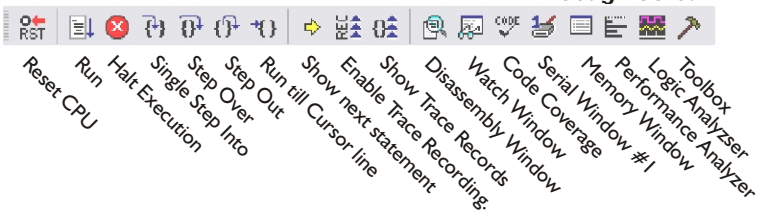
Key	Passes the following item to external tools...
E	Current Editor File.
F	Selected file in the Project Workspace (MEASURE.C).
P	Name of the current project file (MY_APP.UV2).
L	Linker output file, typically the executable file (MY_APP).
H	Application HEX file (MY_APP.HEX).
X	µVision3 executable program file (C:\KEIL\UV2\UV2.EXE).
Q	File name that holds comments for the SVCS system.
R	String that holds a revision number for the SVCS system.
C	String that holds a check point string for the SVCS system.
U	User name specified for the SVCS system.
V	Database file name specified for the SVCS system.

Find in Files *searches text in multiple folders. Click on a search result to open the file in the editor.*



Debugger

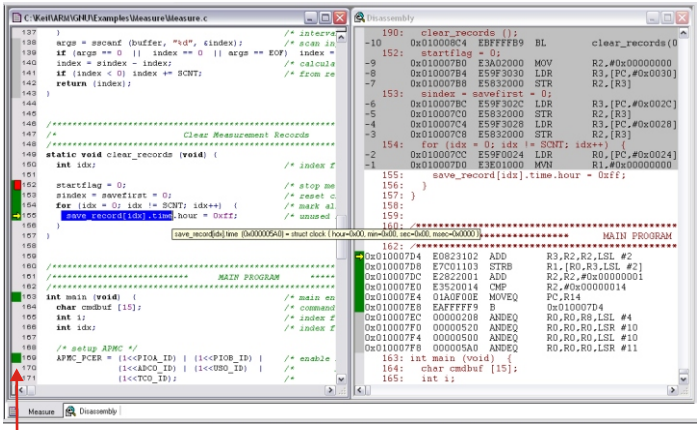
Debug Toolbar



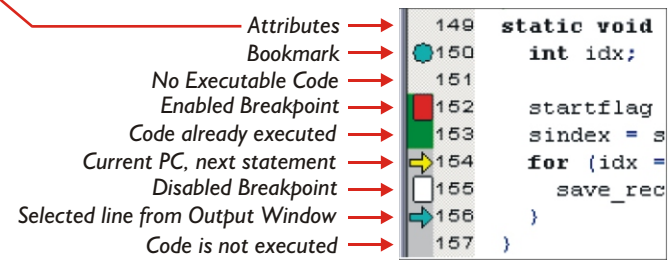
The µVision3 IDE/Simulator/Debugger accelerates your learning efforts by providing a single environment for editing, simulating, and testing target hardware. Most debugger and editor functions may be quickly accessed from the toolbar.

You may use the context menu or the Editor Toolbar to insert breakpoints. Breakpoints you set while editing are activated when you start your debugging session.

µVision3 marks the status of each source line in the **Attributes** column of the editor window. This provides a quick overview of the current breakpoint and execution status.



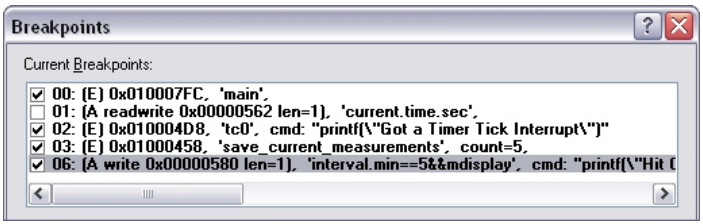
The Disassembly Window includes a trace history that shows the last CPU instructions executed prior to a breakpoint.



Complex Breakpoints

In addition to simple breakpoints, the µVision3 Debugger supports complex breakpoints (with conditional or logical expressions) and memory access breakpoints (with read and/or write access from an address or range).

Complex breakpoints may be defined in the **Breakpoint** dialog (available from the Debug menu). For each breakpoint, you may include a command to execute and a count before the breakpoint is triggered. Commands which are executed when the breakpoint is triggered may even invoke debug functions.

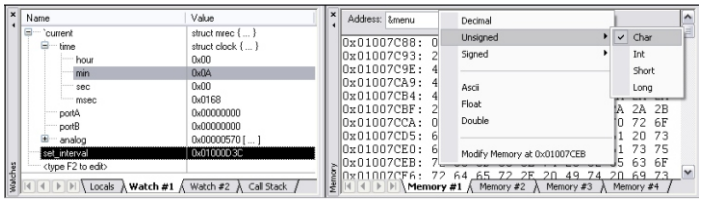


The Breakpoints Dialog allows you to define complex breakpoints on conditional expressions or different memory access types.

Memory and Watch Window

The Memory Window displays various memory areas. You may select the output format or change values via the context menu.

The Watch Window display and allows you to change variables values. The variables of the current functions are shown under the **Locals** tab whereas the **Call Stack** tab lists the program call tree.



The Watch and Memory Windows allow you to view variables and memory. Select display formats with context menu commands.

CPU and Peripheral Simulation

µVision3 has powerful simulation capabilities that give you serious benefits for rapid, reliable embedded software development.

- Simulation allows software testing on your desktop with no hardware environment.
- Early software debugging on a functional basis improves overall software reliability.
- Simulation allows breakpoints that are impossible with hardware debuggers.
- Simulation offers optimal input signals (hardware debuggers add extra noise).
- Single-stepping through signal processing algorithms is possible. External signals are stopped when the CPU halts.
- Failure scenarios that would destroy real hardware peripherals are easily checked.

High-Speed Device Simulation

The µVision3 Debugger accurately simulates a complete microcontroller including the instruction set and on-chip peripherals. Peripheral simulation is composed of:

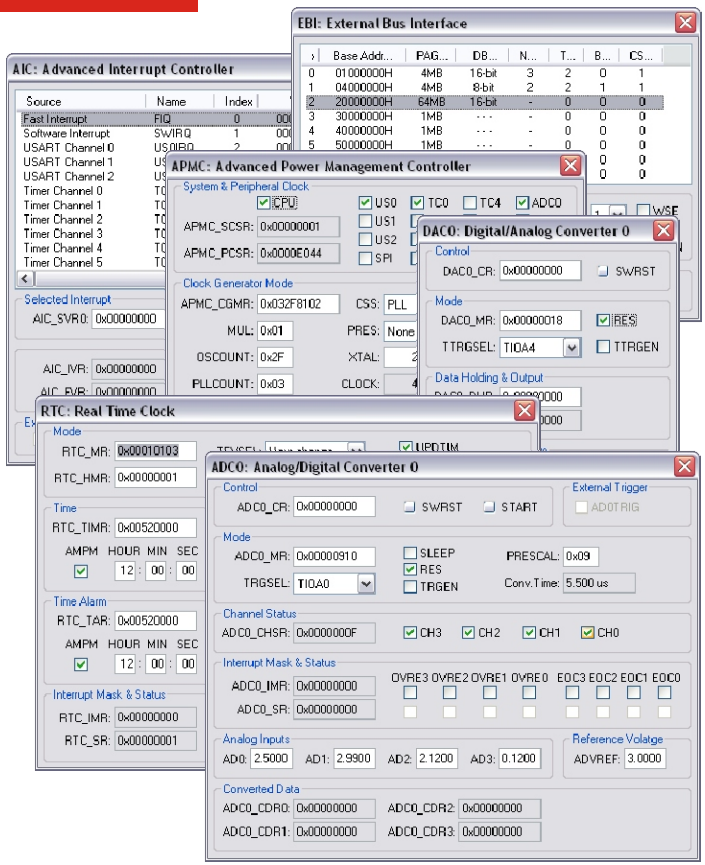
- Dialog boxes that allow you to view and rapidly configure peripherals.
- Virtual Registers (VTREGs) that allow you to control peripherals using breakpoints or the built-in debugging C script language.

You may use dialogs, virtual registers, and debug scripts to create high-fidelity simulations of your target hardware and environment.

The DLL-based Advanced Simulation Interface (AGSI) allows you to implement user-defined peripherals. You may define the behavior and dialogs necessary for a new peripheral. For instance, Dolphin Integration's SUCCESS™ uses AGSI for co-simulation of hardware and software mixed-signal peripherals in VHDL.

Communication Interfaces

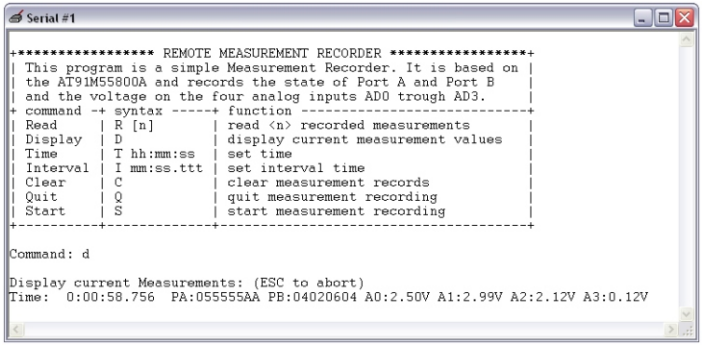
µVision3 supports simulation of I/O interfaces like CAN, I²C, and UARTs. Serial Windows provide terminal simulators for the on-chip UARTs. CAN and I²C Windows show message traffic and bus configuration details.



Peripheral dialogs display detailed status information and aid in driver development.

VTREG	Description
PORTx	Represents the microcontroller I/O port pins.
ADx	Allows to supply a A/D input voltage as float value.
DAx	Shows the D/A output voltage as float value.
CLOCK	Represents the CPU clock frequency
SIN, SOUT	Input/Output registers of the Serial Interface (UART).

The peripheral VTREG's provide access to I/O peripherals from debug scripts, signal functions and the debugger command line.



Serial windows display UART output from your applications. You may type characters to receive into the selected serial window.

Debug Scripts and Signal Functions

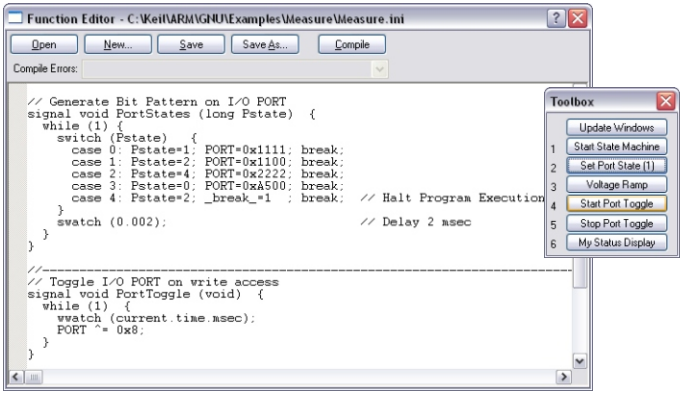
The µVision3 Debugger incorporates a C script language you can use to create:

- Signal functions that simulate analog and digital input to the microcontroller. Signal functions run in the background while µVision3 simulates your target program.
- User functions that extend the command scope or combine repetitive operations.

The µVision3 Debugger includes many built-in functions (like printf, memset, rand, sin, log, sqrt, and so on).

Debug functions may be invoked from the Command Window (by typing the function name), from triggered Breakpoints, or from Toolbox Buttons.

Various debugger output may be directed to log files for long-term storage, analysis, and regression testing.



The Function Editor (Debug menu) allows you to create Signal and User Functions. You may define Toolbox buttons for fast command invocation.

CAN Communication Example

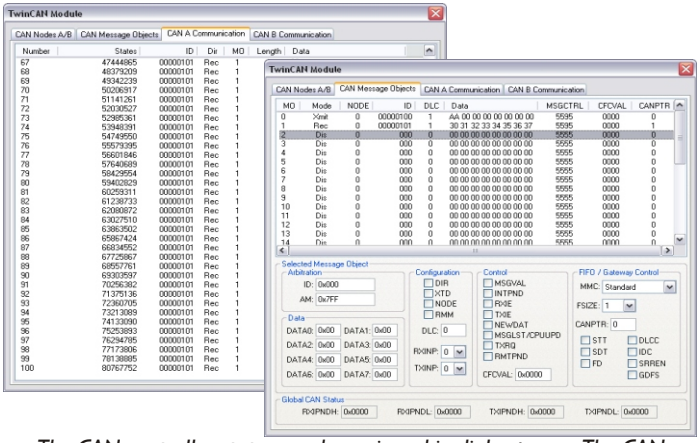
You may use the following VTREGs to access the CAN controller in user and signal functions.

VTREG	Description
CANxID	CAN Message Identifier
CANxL	Length of the CAN Message
CANxBx	Data bytes of the CAN Message
CANxIN	Status of incoming Message
CANxOUT	Status of outgoing Message
CANTIMING	Adjust CAN bus timing

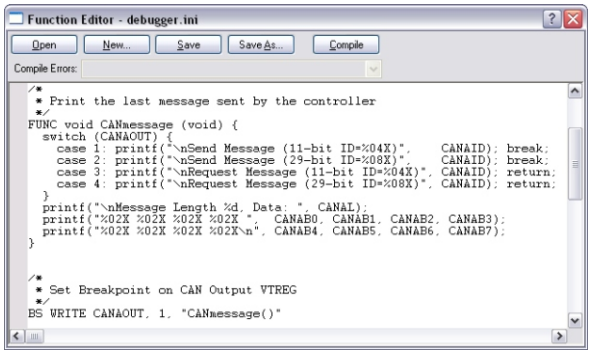
Using these VTREGs you may create a user function that sends a CAN message with 3 data bytes to the application. For example:

```
FUNC void SendMessage (void)  {
  CANAID = 0x4422;  // CAN Identifier
  CANAL  = 3;       // Message Length
  CANAB0 = 0x11;    // 1st data byte
  CANAB1 = 0x22;    // 2nd data byte
  CANAB2 = 0x33;    // 3nd data byte
  CANAIN = 2;       // Send Message
}
```

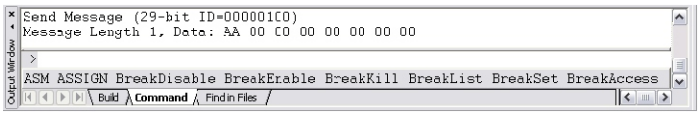
This function may be called from a signal function to repeatedly send this CAN message. You may create user functions that generate automated replies to CAN messages. This is shown in application notes provided at www.keil.com/appnotes.



The CAN controller status can be reviewed in dialog pages. The CAN Communication Page is an analyser which displays all CAN message.



You may use debug scripts to analyse CAN messages. A breakpoint on write access to the VTREG CANxOUT may be used to trigger a user function.



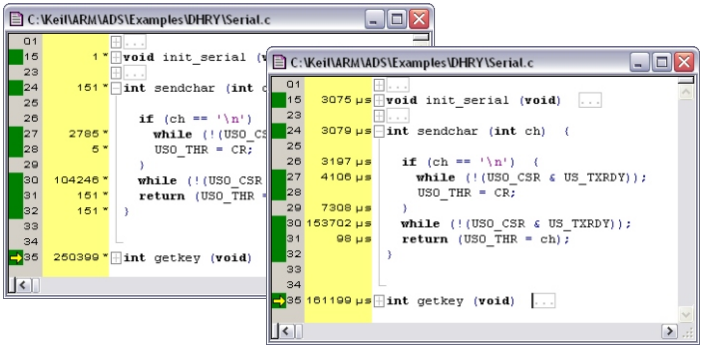
The Output Window displays printf status information of debug scripts.

Advanced Simulation Features

Execution Profiling

µVision3 for ARM provides an execution profiler that records execution statistics for each CPU instruction. You may review the execution count or execution time in editor and disassembly windows. Editor outlining provides an overview of the source code.

You may use the information gathered by the execution profiler to determine where your program spends most of its time. Then, you can concentrate your efforts on making that section run faster.



Execution Profiling shows how many times each line of code has executed and the total execution time.

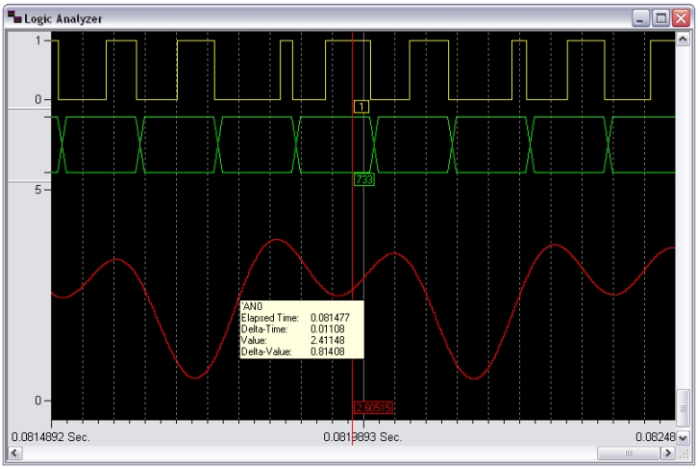
Logic Analyzer

The Logic Analyzer records the values of variables and peripheral I/O signals over time and displays them in the Logic Analyzer Window. The cursor line allows you to set a reference time and review delta values.

The Logic Analyzer provides three different display formats:

- **Bit:** Shows output as a logic level (0 or 1).
- **State:** Shows state changes of a value.
- **Analog:** Shows a graphical display.

You can easily generate complex digital or analog input signal patterns using signal functions you define using the debugger's C script language. This allows detailed testing of signal processing algorithms at simulation level without using extensive hardware equipment.



The Logic Analyzer records signals you select. The above recording shows a PWM output, A/D input, and analog signal generated by a signal function.

```
FUNC float sine_wave (float f)  {
  return __sin (f * (STATES / CLOCK) * 2 * PI);
}

AN0 = sine_wave (2200) + sine_wave (4500);
```

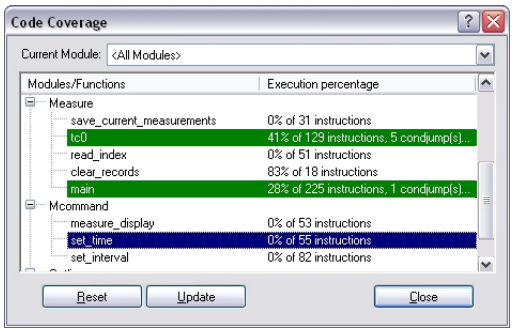
Signal Functions may generate analog and digital signal patterns using peripheral input pins (for example, AN0 is Analog Input 0).

Code Coverage

The µVision3 Debugger provides code coverage statistics for verification of safety-critical applications where certification testing and validation is required. Output includes:

- Non-executed Instructions,
- Skipped Instructions,
- Partially Executed Instructions,
- and Fully Executed Instructions.

Code Coverage information may be output to a file for detailed analysis. You may even include a complete CPU instruction listing in this report file.



Code Coverage provides detailed information about tested functions.

Target Debugging

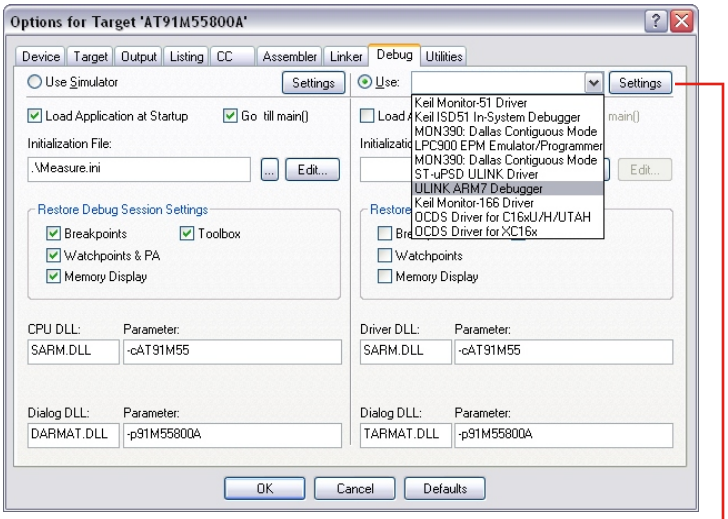
The µVision3 Debugger interfaces to your target hardware via the following Advanced Generic Debugger Interface (AGDI) drivers:

- 8051, 251, and C16x **Monitor**: May be integrated with user hardware. Included with many third-party evaluation boards.
- **ISD51** and **FlashMon**: Designed for 8051 single-chip devices.
- **EPM900**: Philips LPC9xx Emulator.
- **SmartMX DBox**: Philips SmartCards.

Third-party vendors offer the following AGDI target debugger drivers:

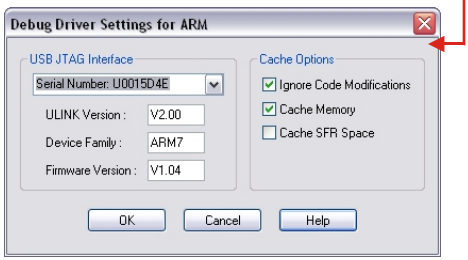
- ChipCon **CC1010** RF Transceiver.
- JTAG interface for **Cygnal 51Fxxx** series.
- Cypress **USB development kits**.
- Infineon **SLE66** SmartCard ROM Monitor.
- Quickcore **FPGA based Pro8051** device.
- SST **SoftICE** for FlashFlex51devices.
- JTAG interface for **Triscend E5** series.

In addition, many emulator vendors offer µVision3 AGDI drivers for their emulators.



µVision3 gives you the choice between Simulator and Target Debugger. Simply select the AGDI driver (in the Options - Debug dialog page) that matches your target hardware configuration.

Click on Settings to open the Driver configuration dialog to specify hardware-related parameters.



ULINK USB-JTAG Adapter

The ULINK USB-JTAG Adapter connects the µVision3 Debugger to the JTAG port pins of the embedded system. Currently ULINK supports ST µPSD 8051-based, Infineon xC16x, and several ARM-powered microcontrollers. ULINK allows you to:

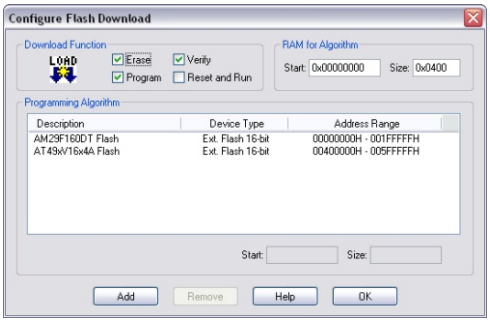
- Download target programs.
- Examine memory and registers.
- Single-step through programs.
- Insert multiple breakpoints.
- Run programs in real-time.
- Program on-chip FLASH.

ULINK connects the PC USB Port to the JTAG or OCDS interface of the microcontroller system and allows Flash programming and debugging.

Flash Device Programming

ULINK supports Flash Device Programming via the µVision3 **Flash - Download** menu. In the **Options - Utilities - ULINK - Settings** dialog you may add programming algorithms for the various Flash devices of your system. ULINK even allows you to customize your own programming algorithms. Complete setup details are saved in the µVision3 project.

Flash programming may be configured for user-provided devices. Several algorithms are available for standard Flash devices and on-chip Flash ROM.



Evaluation Boards

Evaluation Boards and Starter Kits are a great way to begin your microcontroller projects. Keil Software offers evaluation versions of the development tools and example programs which are ready-to-run. You can easily test device performance and the flexibility of the development tools at the same time.

Evaluation boards that interface to µVision3 are available from many chip vendors and thrid-party manufacturers (see the Device Database™ at www.keil.com/dd). In addition, Keil Software offers Development Hardware for the following platforms.

8051 & 251 Boards

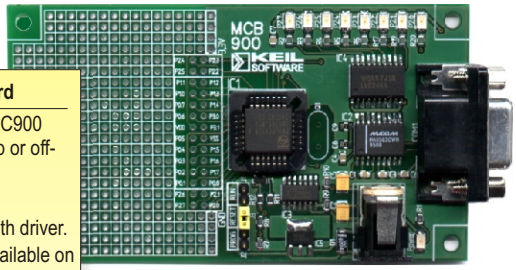
- **MCBx51:** Supports numerous 44-pin PLCC 8051 and 251 devices including many high-speed versions.
- **MCB517AC:** Based on the full-featured Infineon C517A. Includes 81C90 CAN controller and code banking support.
- **MCB900** and **EPM900:** Support the flexible Philips LPC900 low-power devices with a high-speed, two-cycle 8051 core.

XC16x/C16x/ST10 Boards

- **MCB167NET:** Supports all variants of the Infineon C167 and ST10-F168 devices (up to 50MHz CPU speed). It includes two CAN interfaces and an Ethernet controller for TCP/IP applications.
- **MCB-XC167:** Supports Infineon XC16x devices (up to 40MHz) including Flash ROM and MAC unit (for high-speed DSP processing). Supports devices with OCDS (On-Chip Debugging System).

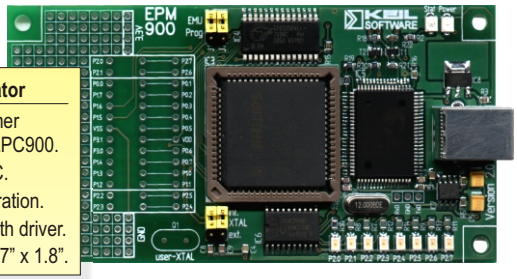
ARM Boards

- **MCB2100:** Supports the ARM7TDMI-based Philips LPC2100 devices and the JTAG debug interface. Members of this device family offer high-speed, on-chip Flash ROM with ISP/IAP programming and include a wide range of on-chip peripherals like A/D converters and communication interfaces like CAN and I²C.



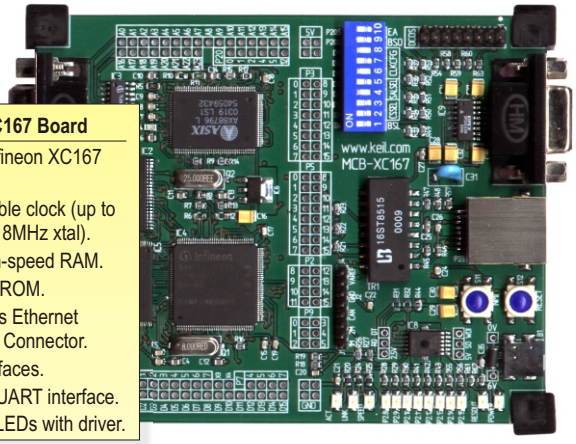
MCB900 Board

- Supports Philips LPC900 devices with on-chip or off-chip clock.
- UART interface.
- Port status LEDs with driver.
- All CPU pins are available on 2.1" x 1.8" prototyping area.



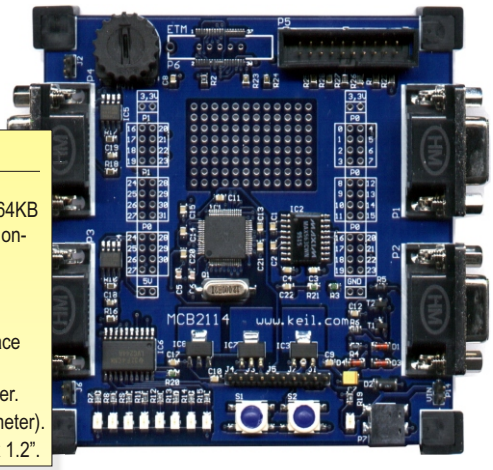
EPM900 Emulator

- Emulator/Programmer Module for Philips LPC900.
- USB interface to PC.
- Flexible clock generation.
- Port status LEDs with driver.
- Prototyping area: 0.7" x 1.8".



MCB-XC167 Board

- Supports Infineon XC167 (XC161).
- PLL for flexible clock (up to 40MHz with 8MHz xtal).
- 512 KB high-speed RAM.
- 2 MB Flash ROM.
- 10/100 Mbps Ethernet Controller & Connector.
- 2 CAN Interfaces.
- OCDS and UART interface.
- Port status LEDs with driver.



MCB2100 Board

- Supports Philips ARM LPC21xx series (up to 64KB RAM and 256KB Flash on-chip).
- 2 CAN interfaces.
- 2 UART interfaces.
- JTAG debugging interface (ETM optional).
- Port status LEDs & driver.
- Analog Input (potentiometer).
- Prototyping area: 0.7" x 1.2".

µVision3: The Intelligent Embedded IDE

Product Highlights

µVision3 reduces your learning curve, shortens the development/testing cycle, and helps you build reliable, stable applications. To accomplish this, µVision3 offers:

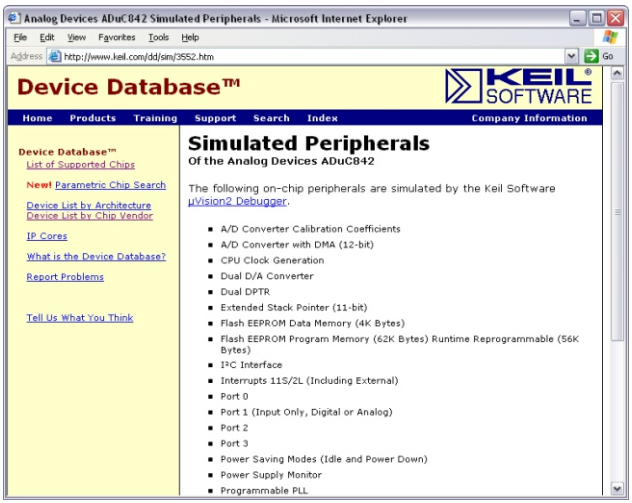
- The Device Database™ (available on Internet and in µVision3) to pre-configure the tools. This minimizes mistakes in project setup and gets you started quickly.
- A full-featured Editor (integrated into the Debugger) that allows source code changes at any time.
- The Debugger which accurately simulates target systems including instructions and on-chip peripherals. Test application code before hardware is available. Optimize your hardware design prior to production.
- Chip-specific dialogs which allow you to review status information and rapidly configure peripherals.
- Identical Simulator and Target Debugger interfaces.
- Integration with third-party tools like SVCS, CASE, and Flash/Device Programming utilities. All configuration details are saved in a single, consistent project.
- The µVision3 Debugger which interfaces to the Keil ULINK USB-JTAG Adapter and other third-party diagnostic tools.

*The Device Database™ on **www.keil.com/dd** provides detailed information about the chips supported by Keil development tools.*

New Device Support

Frequent updates to the Device Database™ ensure that µVision3 always supports the latest 8051, 251, C16x. XC16*x*, ST10, and ARM powered microcontrollers.

Find out more about the Device Database™ and the chips supported! Refer to **www.keil.com/dd**.



Complete peripheral simulation is available for most devices.

Keil Elektronik GmbH and **Keil Software, Inc.**, have representatives in the following countries:

Australia, Austria, Belgium, Brazil, Canada, China, Czech, Denmark, Finland, France, Germany, India, Ireland, Israel, Italy, Japan, Korea, Malaysia, Mexico, Netherlands, New Zealand, Norway, Poland, Portugal, Russia, Romania, Singapore, Slovenia, Slovakia, Spain, South Africa, Sweden, Switzerland, Taiwan, Turkey, Thailand, United Kingdom, and United States.

Europe:

Keil Elektronik GmbH
Bretonischer Ring 15
85630 Grasbrunn
Germany

Phone	**++49 89 / 45 60 40 - 0**
Support	**++49 89 / 45 60 40 - 24**
FAX	**++49 89 / 46 81 62**
Email	**sales.intl@keil.com**
	support.intl@keil.com

United States:

Keil Software, Inc.
1501 10th Street, Suite 110
Plano, Texas 75074
USA

Phone	**800-348-8051**
	972-312-1107
FAX	**972-312-1159**
Email	**sales.us@keil.com**
	support.us@keil.com

KEIL SOFTWARE

www.keil.com

Copyright © 1998 - 2004 Keil Elektronik GmbH & Keil Software, Inc. All rights reserved.
Information in this data sheet is subject to change without notice and does not represent a commitment on the part of Keil Software, Inc.