The Keil C51 ANSI C compiler lets you create C programs for the 8051 microcontroller while retaining the efficiency and speed of hand-optimized assembly. Extensions incorporated into the C51 compiler give you full access to all CPU resources and derivatives. C51 is fully integrated into the µVision2 IDE that combines Compiler, Assembler, Real-Time OS, project manager, and debugger in a single, intelligent environment.

C51 Version 6 is the most efficient and flexible 8051 development tool platform available. With support for all derivatives and full compatibility with emulators and other third party development tools, C51 Version 6 is clearly the best choice for your 8051 projects.

This Brochure Contains Information About:
µVision2 Integrated Development Environment.........................Page 2
C51 Optimizing ANSI C Compiler .............................................Page 4
RTX51 Real-Time Kernel.......................................................Page 6
Program Debugging & Single-Board Computers..........................Page 7
Project Management

A project is composed of all the source files, development tool options, and directions necessary to create a program.

A single µVision2 project can generate one or more Target programs. The source files used to create a target are organized into Groups. Development tool options may be set at the target, group, or file level.

µVision2 includes a Device Database that automatically sets the assembler, compiler, linker, and debugger options required to make full use of the specific microcontroller you choose. The database contains information about on-chip memory and peripherals, and special features such as extra data pointers or math accelerators for virtually every 8051-family microcontroller.

µVision2 can generate the necessary linker options for off-chip memory: just specify the starting address and size.

Integrated Utilities

µVision2 contains powerful features that help you complete your projects on time:

- The integrated Source Browser uses a symbol database for quick navigation through your source files. Use the detailed symbol information to optimize the memory types of your variables.
- Find in Files performs a global text search in all specified files.
- The Tools Menu allows you to start user utilities within the µVision2 IDE.
- A configurable SVCS interface provides access to Version Control Systems.
- The PC-Lint interface gives you advanced syntax analysis of your application code.
- The Infineon EasyCASE interface integrates block level code generation.
- Infineon’s DAvE assists you with CPU and peripheral programming. DAvE projects can be directly imported into µVision2.

The Device Database simplifies tool configuration and provides options that are relevant for the selected CPU.

The Edit, Build, and Debug toolbar gives you quick access to important editor, IDE, and debugger commands.

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

Right-click the mouse to open context sensitive menus in local windows. In the Editor Window, you can access browser information or debugger commands.
Source Code Editor
The µVision2 editor includes all the editing features to which you are accustomed. Color syntax highlighting and text indentation are optimized for C source code. The editor is available while debugging your program. This gives you a natural debugging environment that lets you quickly test and correct your application.

Breakpoints
µVision2 lets you set program breakpoints while editing (even before the source code is compiled or assembled). Breakpoints are activated when you start the µVision2 Debugger and may be set on conditional expressions or variable and memory accesses. Debugger commands or debug functions may be executed when breakpoints are triggered.

The Attributes column provides a quick overview of the current breakpoint settings and the source line status. Code coverage information lets you distinguish between executed and unexecuted program sections.

Debug Function Language
µVision2 incorporates a C-like function language that lets you generate and use:
- Built-in functions like printf, memset, rand, and other useful routines.
- Signal functions to simulate analog and digital inputs to the CPU.
- User functions to extend the command scope and combine repetitive actions.

Variables and Memory
You can point to a variable in the editor to show its value. Two docking windows display and allow the modification of:
- Local variables of the current function.
- User-specified variables in two different pages of the Watch Window.
- Program call tree in the Call Stack page.
- Up to four memory areas each in a different format.
The Keil C51 compiler conforms to the ANSI standard and is designed specifically for the 8051 microcontroller family. Compiler language extensions give you full access to all resources in your embedded application.

**Memory and SFR Access**

The C51 compiler provides complete access to all hardware components of the 8051 family. Access to special function registers is provided by the `sfr` and `sbit` keywords. Variables may be assigned to any address space. With the `_at_` keyword you may even locate variables at fixed memory addresses.

The memory model determines the default memory selector used for variables. However, you may always explicitly specify a memory selector for any variable.

The Linker supports code banking for up to 32 code banks. This lets you expand programs past the 64 KByte ROM space of the 8051. Debugging for code banked applications is fully supported in the µVision2 Debugger and many high-end emulators.

**Interrupt Functions**

C51 lets you create interrupt service routines in C. Fast entry/exit code and register bank switching make C interrupt functions extremely efficient.

Recursive or reentrant functions are defined with the `reentrant` keyword. Functions called from multiple tasks or from interrupt and non-interrupt code must be `reentrant`.

**Flexible Pointers**

C51 provides flexible, efficient pointers. Generic pointers store the memory type and the address of an object in three bytes and may access any variable regardless of its location in the 8051 memory space.

Memory specific pointers are declared with a memory type and always refer to a specific memory area. Since only one or two bytes are stored for the address, pointer accesses are fast.

**C51 Highlights**

- Support for all 8051 derivatives and variants
- Fast 32-bit IEEE floating-point math
- Efficient interrupt functions and direct control of the 8051’s register banks
- Bit-addressable objects
- Sophisticated syntax checking and detailed warning messages
- Use of AJMP and ACALL instructions
- Memory banking support to expand the 64KB limit
- C-level access to all 8051 SFRs
- Register parameters and variables with global register optimization
- Use of multiple data pointers
- Use of on-chip arithmetic units
- Support for Philips 8xC750, 751, 752 limited instruction sets
- Generic and memory-specific pointers
- Reentrant functions and register bank independent code
- Debug information support for all emulator vendors
- Simple assembly language interface
- Fully compatible with object files generated by Intel ASM51
- Easily interfaced to Intel PL/M-51 programs

**Memory Selectors**

<table>
<thead>
<tr>
<th>Selector</th>
<th>Address Space</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>data</code></td>
<td>128 Bytes directly addressable on-chip RAM, fastest access; default space in SMALL Memory Model.</td>
</tr>
<tr>
<td><code>bdata</code></td>
<td>16 Bytes bit-addressable RAM; mixed bit and byte access.</td>
</tr>
<tr>
<td><code>idata</code></td>
<td>256 Bytes indirectly addressable on-chip RAM.</td>
</tr>
<tr>
<td><code>pdata</code></td>
<td>256 Bytes paged external RAM; default in COMPACT Model.</td>
</tr>
<tr>
<td><code>xdata</code></td>
<td>64 KB external RAM; default in LARGE Memory Model.</td>
</tr>
<tr>
<td><code>code</code></td>
<td>64 KB program memory.</td>
</tr>
</tbody>
</table>

**Memory Specific and Generic Pointers**

<table>
<thead>
<tr>
<th>Pointer Type</th>
<th>idata Pointer</th>
<th>xdata Pointer</th>
<th>generic Pointer</th>
</tr>
</thead>
<tbody>
<tr>
<td>C Source</td>
<td>char idata *ip;</td>
<td>char xdata *xp;</td>
<td>char *p;</td>
</tr>
<tr>
<td>val = *ip;</td>
<td>val = *xp;</td>
<td>val = *p;</td>
<td></td>
</tr>
<tr>
<td>Code Generated</td>
<td>MOV R0,ip</td>
<td>MOV DPH, xp+1</td>
<td>MOV R1.p+2</td>
</tr>
<tr>
<td>MOV val, @R0</td>
<td>MOV SPL, xp</td>
<td>MOV R2.p+i</td>
<td></td>
</tr>
<tr>
<td>MOV DPH, @A</td>
<td>MOV A, @DPT</td>
<td>MOV R3.p</td>
<td></td>
</tr>
<tr>
<td>MOV val, A</td>
<td>CALL CLDPR</td>
<td>MOV CALL</td>
<td></td>
</tr>
<tr>
<td>MOV @DPT, val</td>
<td>MOV CLDPR, A</td>
<td>MOV CLDPR, val</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Pointer Size</th>
<th>CPU Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Byte</td>
<td>4 cycles</td>
</tr>
<tr>
<td>2 Bytes</td>
<td>7 cycles</td>
</tr>
<tr>
<td>3 Bytes</td>
<td>13 cycles</td>
</tr>
</tbody>
</table>

*C51 supports both generic and memory specific pointers.*
General Code Optimizations
- Constant Folding
- Common Subexpression Elimination
- Strength Reduction
- Control Flow Optimization
- Register Variable Usage
- Parameter Passing in Registers
- Loop Rotation
- Dead Code Elimination
- Common Tail Merging
- Common Block Subroutine Packing

8051-Specific Optimizations
- Peephole Optimization
- Switch-Case Optimization
- Interrupt Function Optimization
- Data Overlaying
- Extended Access Optimization

Register Optimization
C51 allocates up to 9 CPU registers for function arguments and local variables. Up to three arguments may be passed in registers. Project-wide global register optimization removes unnecessary code and optimizes CPU register utilization.

C Run-Time Library
The C51 run-time libraries consist of over 100 functions, most of which are reentrant. The libraries support all of the ANSI C routines that are consistent with the limitations of embedded applications.

Intrinsic routines are provided for hardware-specific instructions like `nop, testbit, rol,` and `ror`.

A51 Macro Assembler
The A51 Macro Assembler supports both standard and MPL macros. To quickly create assembly routine shells, use the C51 compiler’s `SRC` directive.

New in Version 6, A51 lets you include your C header files to define constants and to declare special function registers. Now, a single header file can be used in C and assembly programs.

C51 Version 6: New Optimization Levels
C51 Version 6 offers three new optimizer levels beyond those in C51 Version 5. The new optimizations focus primarily on code density and typically reduce program size by 10%. If your application is 35KB with C51 Version 5, it will most likely fit into a 32KB ROM when you re-compile with C51 Version 6 and Optimizer Level 9.

Optimizer Level 7: Extended Access Optimization
Optimizer Level 7 uses the DPTR for register variables where appropriate. Pointer and array accesses are optimized for both execution speed and code size.

```
struct s { char a; char b; };
struct s xdata sarray[100];
unsigned char search(char v) {
    struct s *sp;
    unsigned char i;
    sp = sarray;
    for (i = 0; i != 100; i++) {
        if (sp->a == v) return (i);
        sp++;
    }
    return (0xFF);
}
```

```
Example Program

<table>
<thead>
<tr>
<th>Code Size</th>
<th>V5.5</th>
<th>V6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>31</td>
<td>21</td>
</tr>
</tbody>
</table>
```

Optimizer Level 8: Reuse Common Entry Code
When there are multiple calls to a single function, some of the setup code can be reused, thereby reducing program size.

```
int func (int i) {
    return (i + 10);
}
```

```
int xdata il, xdata i2;
int vl, v2;
void main (void) {
    vl = func (il++);
    v2 = func (i2++);
}
```

```
Optimize 8 is the new default level of C51 Version 6.

<table>
<thead>
<tr>
<th>Code Size</th>
<th>V5.5</th>
<th>V6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>49</td>
<td>39</td>
</tr>
</tbody>
</table>
```

Optimizer Level 9: Common Block Subroutines
C51 detects recurring instruction sequences and converts them into subroutines. C51 even rearranges code to obtain larger recurring sequences. This optimization is most effective on large modules.

```
int xdata array[100];
int j1, j2;
int xdata il;
extern void func (void);
void main (void) {
    j1 = array[il+j2];
    func (j2 = array[il+j2]);
}
```

```
<table>
<thead>
<tr>
<th>Code Size</th>
<th>V5.5</th>
<th>V6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>82</td>
<td>55</td>
</tr>
</tbody>
</table>
```

```
The RTX51 multitasking real-time kernel makes implementing complex, time-critical software projects easy. RTX51 is royalty-free and is fully integrated into the Keil C51 tool chain supporting both C and assembly language programs. RTX51 fully supports code banking applications.

**RTX51 Full** supports:
- Standard tasks (with a shared register bank),
- Fast tasks (with dedicated register banks),
- CAN (Controller Area Network),
- Mailboxes (for inter-task communication),
- Memory pools.

**RTX51 Tiny** is a reduced functionality kernel that fits on single-chip systems.

Both let you create and delete tasks, send and receive signals, and perform other operating system management.

### Context Switching

- **Round-Robin**: Each task runs for a defined time slice. When a task’s time slice is up, the next task starts.
- **Preemptive**: The task with the highest priority runs until it is preempted by a higher priority task or until it delays waiting for an event.

### Events and Interrupts

RTX51 supports the following events:

- **Timeouts**: Delay for specified timer ticks.
- **Intervals**: Delay for an interval of time.
- **Signals**: Coordinate inter-task operations.
- **Messages**: Pass messages between tasks.
- **Interrupts**: Handle hardware interrupts.
- **Semaphores**: Share limited resources.

Interrupt functions are supported and may send signals and messages to and from tasks.

### CAN Interface

RTX51 Full includes a library of routines for CAN. The following devices are supported: Infineon 81C90/91, C505C, and C515C; Intel 82526 and 82527; and Philips 82C200, 8xC592, and 8xCE598.

---

**Overview of RTX51 Routines**

- **Task Management Routines**: create-task, delete-task, and running-task-id.
- **Interrupt Management Routines**: attach-interrupt and detach-interrupt.
- **Signal Management**:
  - send-signal, clear-signal, and isr-send-signal.
- **Semaphore Functions**
  - send-token.
- **Mailbox Functions**: send-message, recv-message, isr-send-message, and isr-receive-message.
- **Memory Management Functions**
  - create-pool, get-block, and free-block.
- **System Clock Routines**: timer-ticks and set-slice.
- **Debug Routines**: task-create, hw-init, def-object, start, send, write, receive, def-buf-size, bind-obj, unbind-obj, wait, request, read, and get-status.

---

**Technical Data**

<table>
<thead>
<tr>
<th>RTX51 Full</th>
<th>RTX51 Tiny</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of tasks</td>
<td>256 max., 19 active</td>
</tr>
<tr>
<td>Number of mailboxes</td>
<td>8 max with 8 entries each.</td>
</tr>
<tr>
<td>Number of semaphores</td>
<td>8 x 1 bit max.</td>
</tr>
<tr>
<td>RAM Requirements</td>
<td>40-46 Bytes data</td>
</tr>
<tr>
<td>Hardware Requirements</td>
<td>One on-chip timer</td>
</tr>
<tr>
<td>System Clock</td>
<td>1,000 - 40,000 cycles</td>
</tr>
<tr>
<td>User task priorities</td>
<td>0-127</td>
</tr>
<tr>
<td>Context switch time</td>
<td>70-100 cycles (fast task)</td>
</tr>
<tr>
<td>Interrupt lockout time</td>
<td>&lt; 50 cycles.</td>
</tr>
</tbody>
</table>

Available only in RTX51 Full.
The µVision2 Debugger provides all the usual source-level, symbolic debugging features and includes trace history, code coverage, and complex breakpoints. The DDE interface and a script language support automated program testing.

**CPU & Peripheral Simulator**

µVision2 integrates high-speed simulation of the CPU and on-chip peripherals for the 8051 and its derivatives. I/O values may be reviewed and modified directly in dialog boxes or you may write signal functions in the built-in C-like macro language to provide dynamic input.

**Target Monitor**

µVision2 comes with a configurable Monitor that lets you test your software in target hardware. The Monitor works directly with the µVision2 Debugger and supports code banking. The Monitor requires just 6 bytes of stack space, 6KB code ROM, and 256 bytes of xdata RAM in your target system.

**MCB517/251 Starter Kits**

An MCB Starter Kit is a great way to get started with your 8051 project. Each Starter Kit includes a 2K-limited set of development tools and numerous sample programs which are ready-to-run. You can easily test the performance of the 8051 and the flexibility of our development tools at the same time.

The MCB517AC board includes the high-performance Infineon C517A microcontroller that provides standard 8052 peripherals plus A/D converter, PWM, capture/compare, eight data pointers, and a high-speed math unit. The 81C90 CAN controller and code banking support are also included. All these features are fully supported by the Keil 8051 development tools.

The MCB251 board supports all 44-pin PLCC variants of the 8051 family (including high-speed devices) and 251 family. Use this board to evaluate the best 8051 or 251 device for your application.
Advantages of C51 and Related Tools

- New optimizations in C51 generate code that approaches the efficiency of hand-tuned assembly.
- C51 supports all 8051 variants and provides access to all hardware components.
- C51 generates very fast interrupt code with or without 8051 register bank switching.
- C51 supports multiple data pointers and high-speed arithmetic units on the extended 8051 derivatives from Atmel, Dallas, Infineon, Philips, and Temic.
- C51 assigns register variables and performs global register optimization on an entire application.
- All tools generate detailed warning and error messages to help you locate hard-to-find problems.
- C51 supports reentrant functions and register bank independent code for interrupt service routines and multitasking applications.
- Efficient memory banking and debugging let you easily expand the 64 KByte space limit.

Advantages of µVision2

- Easy-to-use integrated development environment provides intelligent setup for tool options.
- High-speed CPU and peripheral simulator with drivers for numerous 8051 derivatives.
- C-like language for creating user and signal functions.
- Integrated performance analyzer and code coverage.
- Interfaces to a configurable target monitor.

12 Years of Keil C51

Since its market introduction in 1988, the Keil Software C51 Compiler has become the leading C compiler for the 8051 microcontroller family. No other 8051 compiler generates more efficient code for your embedded applications. That’s why more than 30,000 users worldwide trust the Keil advantage for their 8051 projects.

Now, C51 Version 6 sets new standards for code density. Compared to C51 Version 5 the new optimization levels of C51 Version 6 reduce the code size of your applications by 5% to 15%.

C51 Compiler: Module Code Size Comparison

<table>
<thead>
<tr>
<th>Code Size</th>
<th>V4.0</th>
<th>V5.5</th>
<th>V6.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dhrystone 1.1</td>
<td>1912 Bytes</td>
<td>1905 Bytes</td>
<td>1776 Bytes</td>
</tr>
<tr>
<td>Whetstone (Single Precision)</td>
<td>4306 Bytes</td>
<td>4336 Bytes</td>
<td>3256 Bytes</td>
</tr>
<tr>
<td>DES Algorithm Example</td>
<td>5849 Bytes</td>
<td>5889 Bytes</td>
<td>4377 Bytes</td>
</tr>
<tr>
<td>MEASURE.C Example</td>
<td>915 Bytes</td>
<td>916 Bytes</td>
<td>880 Bytes</td>
</tr>
</tbody>
</table>

C51 Optimizer Level 9 reduces the code size of the above test programs by 4 to 25%.

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

Australia, Austria, Belgium, Brazil, Canada, Czech Republic, China, Denmark, Finland, France, Germany, Hong Kong, India, Ireland, Israel, Italy, Japan, Korea, Netherlands, Norway, Poland, Portugal, Russia, Singapore, Slovenia, Spain, South Africa, Sweden, Switzerland, Taiwan, United Kingdom, and United States.

Contact Keil Elektronik GmbH or Keil Software, Inc., for the distributor nearest you.

Evaluation software and detailed technical information are available from your distributor and our World Wide Web site (www.keil.com).

European:

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, TX 75074
USA
Phone 800-348-8051
Support 972-312-1107
FAX 972-312-1159
Email sales.us@keil.com
support.us@keil.com

www.keil.com

Copyright © 1992-2000 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.